

UNCLASSIFIED

AD-A252 535
Copy 47 of 50 copies
2

AD-A252 200



IDA PAPER P-2490

DTIC
ELECTED
JUN 12 1992
S C D

**A GENERIC AND TARGET ARCHITECTURE FOR
COMMAND AND CONTROL INFORMATION SYSTEMS**

Richard L. Wexelblat, *Task Leader*

September 1991

Prepared for
Defense Information Systems Agency

Approved for public release, unlimited distribution: 27 February 1992.

92-15351



INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

UNCLASSIFIED

IDA Log No. HQ 90-036067

92 6 11 069

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsor or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 89 C 0003 for the Department of Defense. The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

This Paper has been reviewed by IDA to assure that it meets high standards of thoroughness, objectivity, and appropriate analytical methodology and that the results, conclusions and recommendations are properly supported by the material presented.

© 1991 Institute for Defense Analyses

The Government of the United States is granted an unlimited license to reproduce this document.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 1991		3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE A Generic and Target Architecture for Command and Control Information Systems		5. FUNDING NUMBERS MDA 903 89 C 0003 Task T-S5-771		
6. AUTHOR(S) Richard L. Wexelblat et al.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Institute for Defense Analyses (IDA) 1801 N. Beauregard St. Alexandria, VA 22311-1772		8. PERFORMING ORGANIZATION REPORT NUMBER IDA Paper P-2490		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) JIEO/TVCF Defense Information Systems Agency Center for C3 Systems 3701 N. Fairfax Dr. Arlington, VA 22203		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, unlimited distribution: 27 February 1992.		12b. DISTRIBUTION CODE 2A		
13. ABSTRACT (Maximum 200 words) <p>The Defense Information Systems Agency (DISA) has been directed to upgrade the Worldwide Military Command and Control System (WWMCCS) by providing a Command and Control Information System (CCIS) which offers a combination of services and products that aid interoperability and that have sufficient flexibility to accommodate the varied missions of WWMCCS users. The architectural premise is of an open, interoperable, and heterogeneous set of systems and components linked together to accomplish command and control functions. DISA asked IDA to develop a generic, standards-based architecture consistent with CCIS needs, an architecture based on accepted standards that foster interoperability, evolvability, and allow for technology growth with commercial off-the-shelf products. This paper describes the military requirements, the generic architecture, and the standards profile appropriate for a CCIS to be built in the 1995-97 time period.</p>				
14. SUBJECT TERMS Command and Control Information System (CCIS); Architecture; WWMCCS; WAM; Interoperability; COTS; Interface Standards.		15. NUMBER OF PAGES 446		
		16. PRICE CODE		
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR	

UNCLASSIFIED

IDA PAPER P-2490

**A GENERIC AND TARGET ARCHITECTURE FOR
COMMAND AND CONTROL INFORMATION SYSTEMS**

Richard L. Wexelblat, *Task Leader*

**Earl A. Alluisi
James Baldo
Gregory A. Corliss
Karen D. Gordon
Steve A. Lawyer
Reginald N. Meeson
Beth Springsteen**

**Cy D. Ardoin
John M. Boone
William L. Greer
David S. Hough
Catherine W. McDonald
Richard P. Morton
Stephen R. Welke**

**Edward Bautz
Bill R. Brykczynski
Deborah Heystek
Robert J. Knapper
Terry Mayfield
Edgar Sibley
Christine Youngblut**

September 1991



Approved for public release, unlimited distribution: 27 February 1992.



INSTITUTE FOR DEFENSE ANALYSES

**Contract MDA 903 89 C 0003
Task T-S5-771**

UNCLASSIFIED

Accession For	
NTIS GRAFI	
DTIC TAB	
Unpublished	
Justification	
By _____	
Distribution/ _____	
Availability Codes _____	
Dist	Avail and/or Special
A-1	

PREFACE

This paper reports the results of a study undertaken in response to Task Order T-S5-771, Worldwide Military Command and Control System Automated Data Processing Modernization (WAM) Target Architecture. It contains a definition of a generic architecture for future command and control information systems (CCIS) and of a target profile for 1995-97. An architecture was developed that will ensure that WAM improvements conform to a common set of interface standards consistent with CCIS needs of interoperability, evolvability, and technology growth, using, wherever possible, commercial off-the-shelf products. Military, federal, and national and international industry standards were considered for their applicability to the target architecture during its 1995-1997 timespan. Standards currently in draft were also included if they are relevant and expected to become final during this time period.

The paper is aimed at the following audiences:

- a. Defense Information Systems Agency (DISA) executives performing long-range planning for CCIS programs, including WAM. Most of the recommendations are directed at this level.
- b. Current CCIS program managers guiding procurement decisions that need to be made in the execution of their programs.
- c. CCIS developers making design decisions.
- d. CCIS users anticipating and selecting future options.

This paper was reviewed by the following staff members of the Institute for Defense Analyses: Mr. William Akin, Prof. Thomas C. Bartee, Dr. James Carlson, Dr. Dennis W. Fife, Dr. Harlow Freitag, Ms. Audrey A. Hook, Dr. Richard J. Ivanetich, Dr. Robert P. Walker, and Dr. Robert I. Winner. Mr. Terry Courtwright served as an external reviewer. Their contributions are gratefully acknowledged.

Appreciation is also expressed to Dr. Larry Reeker and Dr. Norm Howes of IDA, Dr. James P. Pennell, formerly of IDA, Dr. Ravi Sandhu of George Mason University, and Dr. Steven R. Schach of Vanderbilt University for their guidance and technical assistance.

**THIS
PAGE
IS
MISSING
IN
ORIGINAL
DOCUMENT**

EXECUTIVE SUMMARY

In response to a request from the Defense Communications Agency (DCA),* the Institute for Defense Analyses (IDA) has developed a generic architecture for command and control information systems (CCIS). The architecture is to cover a 20-year time span. The results of the study include the generic architecture and a target profile listing standard interfaces and protocols appropriate to a system to be built in the 1995-97 time period. IDA also investigated the status of standards and the progress of standardization in areas relevant to the design of a CCIS that conforms to the generic architecture.

Background

WWMCCS, the Worldwide Military Command and Control System, provides the means for operational direction and technical administrative support in the command and control (C²) of U. S. military forces. The work presented in this report is part of the WAM (WWMCCS Automated Data Processing (ADP) Modernization) program. The thrust of the effort is evolutionary, gradually replacing WWMCCS ADP proprietary components by systems that use standard, open, nonproprietary interfaces and protocols. In November 1989, the Director, DCA, approved a Decision Coordinating Paper for the WAM Program, that included the following guidance:

DCA will develop and evolve an architecture based upon the study of user requirements and the solutions available commercially to implement an open systems architecture based upon federal, national and international standards. This architecture will conform to the command and control architecture for command centers... An Architecture Report, including transition plans to the target architecture, will be produced by FY 93 to support continuation of JOPES [Joint Operations Planning and Execution System] development in the post FY 94 period.

IDA was selected to develop the architecture; transition and short-term development were assigned to other contractors. Two additional documents have been prepared from the current year's work: a detailed survey of standards relevant to the design of a military CCIS and a white paper discussing issues of prototyping and validation of the architecture. The standards survey also considers areas where standards are needed but lacking.

* During the period of this study, the name of the sponsoring agency was changed from DCA to DISA, the Defense Information Systems Agency. In the text of the report, "DCA" is used when referring to events occurring prior to the name change.

Requirements for a CCIS

A key hypothesis of this study is that the networked computer systems that best support military command and control bear great similarity to non-military information systems. Thus, opportunities for use of commercial software are excellent. In the best judgement of the development team, this hypothesis is valid.

The essential mission for a military CCIS is to support a commander in exercising command and control of military forces, giving direction to subordinate commands, and receiving critical, time-sensitive warning and intelligence information, thereby ultimately supporting the National Command Authorities. Planning, decision making, and execution characterize the operations of a CCIS. Core functions include threat identification and assessment, strategy determination, development of courses of action, detailed planning, implementation of planned actions, and monitoring. The operation of a CCIS is dependent on timely information reaching the proper person in the proper format at the proper time. Automated techniques for analysis of available information and for simulations based on this information will apply equally to peacetime intelligence, crisis planning, and wartime operations.

The architecture addresses problems with the current system, including reliance on proprietary hardware, fragmentation of data among different applications, lack of flexibility in providing new functionality, disproportionate cost increases at critical points, reliance on obsolete hardware and software, non-standard interfaces, performance bottlenecks that inhibit the use of simulation, unfriendly and inconsistent user interface, rigid mechanisms for executing planning functions, and inefficient and inflexible use of computer resources. The architecture will address these deficiencies by standardizing interfaces, by distribution of computing resources within and among command centers, and by use of commercial off-the-shelf products.

Architectural Principles

An architecture provides the structure of a system, whereas a target profile lists standards used in building a system that conforms to the architecture. The CCIS generic architecture and WAM target profile will promote the following principles:

- a. Systems will be open. People and computer programs will be able to exchange data and services among different parts of the system without regard to differences in suppliers of hardware and software.

- b. Systems will be distributed. Computers and storage devices will be located in different command centers. Command centers may be made up of dispersed cells; key users will carry their computing resources wherever they go. Application programs and data will reside on computers at thousands of locations.
- c. Distributed components will cooperate according to standard protocols that implement, among other things, a client-server model. That is, application programs make use of other programs, possibly on other computers, to perform such services as data management and communications.
- d. Software designs will be layered to encapsulate design details that are expected to change. Three special cases of layered design are the Open Systems Interconnection (OSI) model, separation of data management from applications, and standardization of interfaces to operating systems.

The target profile includes standards for an information system whose interface specifications comply with open system standards defined by the National Institute of Standards and Technology (NIST) and by other standardization groups. The definition is not constrained by the limitations of the present WWMCCS, but contains the best technology expected to be available in the latter half of the present decade. It enables a move toward the goal of an open system while satisfying WAM requirements. Though transition was not part of the present IDA mission, it must be addressed in any plan to implement the architecture.

The architecture provides support for evolvability and interoperability. User needs cannot be predicted with complete accuracy. Requirements change, technology becomes obsolete, and the market continually provides better products. Systems evolve, not in distinct generations, but in a steady flow of changes and improvements. The open system environment supports evolution by separating functions into a layered architecture whose boundaries adhere to accepted standards defined through open consensus. If physical and logical boundaries conform to standards, getting usable information becomes easier. Increasing participation of vendors in developing OSI products confirms that an open systems approach is not only feasible but will become the norm.

Overview of the Generic Architecture

A CCIS is a distributed information system architecture of nodes connected by networks. Node architecture is based on the structure of present-day C² command centers and information processing sites and anticipates a growing need for computing power and

decentralization. Within a node, functions now served by a mainframe computer and terminals will be provided by a local network of servers and workstations. Each node will be connected to others through a wide area network. The CCIS of the future will be a group of interconnected wide area networks. Thus, the generic architecture is a network of networks that connect nodes, each of which may itself contain a network. To describe this complex structure, seven areas are defined to group relevant protocols and interface standards:

- a. *Data Management* supports the storage, control, distribution, management, and allocation of simple data such as text and numeric information and complex items such as complete documents, maps, charts, images, and multimedia objects. Information about the network and nodes themselves (e.g., configuration data) will be also handled through data management services.
- b. *Network Services* are the transmission and interface standards and protocols that support logical and physical communication. They describe and constrain how the hardware and software of the nodes cooperate in node-to-node interaction.
- c. *User Interface Services* support visual and functional interaction with the user, providing access to hardware and software and graphical user interface. They control the presentation format of data and mode of interaction.
- d. *Operating Systems* manage hardware and software resources and program interfaces, including local and distributed execution of application programs. Standards in this domain include those that cover program-to-program communication and synchronization as well as management of memory and interfaces to network and data management services.
- e. *Security* provides for the privacy, protection, and integrity of the programs and data that make up the CCIS. Security pervades the model, applying at every interface and point of data transfer.
- f. *Programming Services* control CCIS application development and the execution of applications.
- g. *Data Exchange Standards* permit the exchange of data among applications, transfer of data between systems, and display of data to the user in a way that preserves the meaning and relationships in that data.

The functional parts of the architecture and the service areas fit together as shown in Figure ES-1, the full derivation of which can be found in Section 3 of this report. Briefly, the architecture has data management, networks, operating systems, user interfaces, and application functions. The equivalent services jointly support the execution of applications. Data Exchange is not associated with a particular part of the structure. It is a set of standards

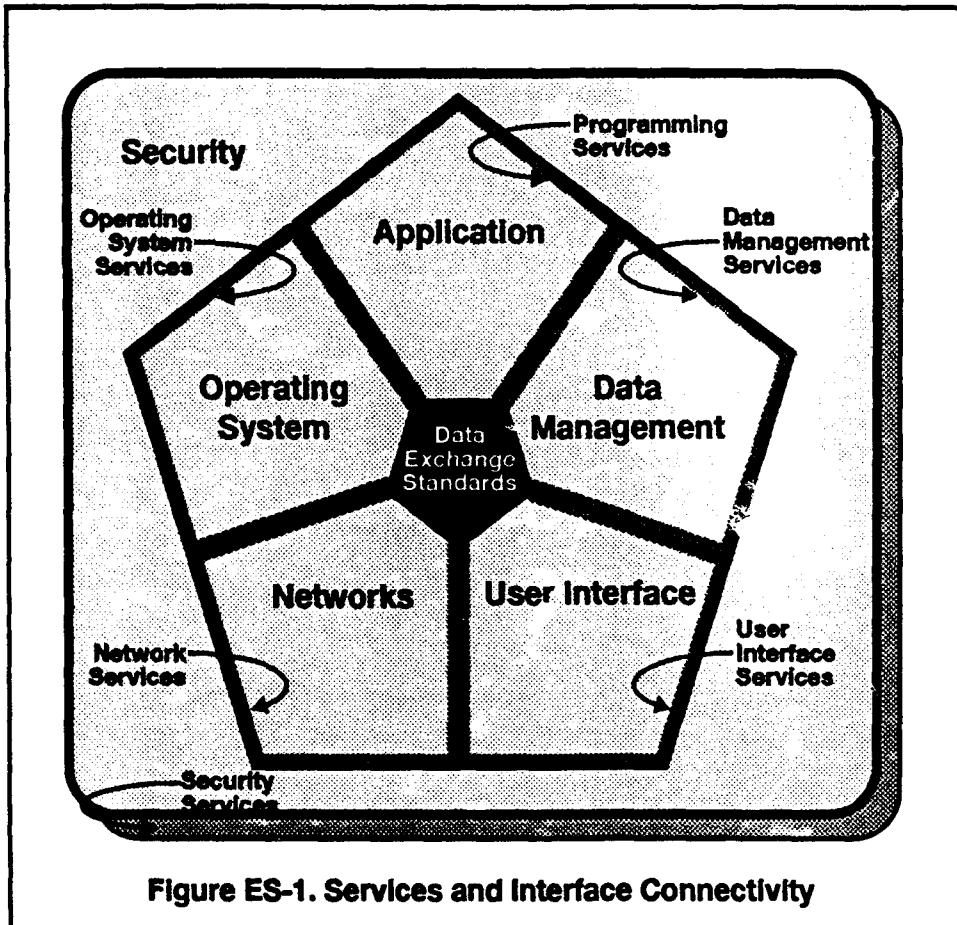


Figure ES-1. Services and Interface Connectivity

that mediate data interchange among applications and between the support functions as well. Security is similarly not associated with a particular part of the structure but can be considered analogous to a “substrate” that underlies the entire structure.

The Target Profile

The target profile is a listing of standards intended to provide guidance to those who will design a CCIS in the 1995-97 time period. It reflects standards expected to be current during that period and identifies areas where standards may be insufficient. Table ES-1 summarizes the specific areas covered by the profile. A more detailed overview appears in Section 4. Appendices A through G cover the standards in significantly more detail. Emerging standards, de facto standards, and occasionally relevant proprietary standards are mentioned where appropriate. Standards are continually updated or under development and it may be to DISA's advantage to participate actively in selected standardization groups.

Table ES-1. The Target Profile

Data Exchange		Operating Systems
X.12, EDIF—Formatted documents		POSIX
ISO 8613, 8839, 10179—Unformatted		Basic services
ISO 8632—Computer graphics files		Ada bindings
ISO 10303—Product model data		Real-time extensions
STDS, VPS, DIGEST—Maps and geographic information		Security extensions
Multimedia		Shells and utilities
ISO 19018, 11172, JBIG, X.56—Data compression		System management
ASCII		Distributed system services
Data Management		Programming Services
Data management reference model		Ada
SQL		C and C++
Information resources dictionary system		Other languages
Remote database access		CAIS-A
Distributed transaction processing		PCTE+
Networks		Security Services
ISO-OSI reference model		“Rainbow books”
GOSIP		Trusted database management
Upper layer standards		Secure data network system
Lower layer standards		Standard for interoperable LAN security
Intermediate layer standards		
Application layer standards		
Directory services		User Interface
FTAM		User interface architecture (based on APP)
Message handling		Window management
Remote data access		Toolkit and user interface management
Remote procedure call		Terminal management
Transaction processing		Graphics
Telematic standards		Graphics subroutine library
Virtual terminal		Device independent and dependent stds.
		Color standards
		Rendering standards
		Human factors standards
Note: The details of the Target Profile (including definition of all abbreviated terms) may be found in Section 4.		
The NIST APP (Applications Portability Profile) is basic to the Target Profile. It appears explicitly or implicitly in most of the service areas.		

Although multi-level security is part of the given CCIS requirement, it is not likely that standards for validated, trusted multi-level security will be in place in time for a 1995-97 development. DISA is actively working on the security issue. In July of 1990, the Director of the Defense Communications Agency established a joint DCA-NSA “Defense-wide

Information Systems Security Program." Pending the report on this activity, no recommendations for security beyond system-high mechanisms are made in the target profile.

Discussion and Recommendations

During course of this study, certain observations were made and recommendations based on these observations are provided. The most significant of these are summarized here. Additional discussion and further recommendations appear in Section 5.

Recommendation. The architecture needs to be tested against a broad sample of requirements to ensure that it is appropriate to all CCIS applications.

The present activity considered only those requirements developed for the WAM program. The architecture has broad scope, hence a broader look at requirements is in order.

Recommendation. The contention that the architecture is not different in any substantial way from what is required of non-C² distributed information systems should be examined in order to ensure that the most appropriate set of standards and profiles are selected for CCIS use.

Two aspects need to be considered. First, if commercial directions are ignored, substantial cost savings and other benefits could be lost. Second, if the commercial lead is followed blindly, significant CCIS requirements could go unmet. Neither case is acceptable.

Recommendation. CCIS standards should be commercial standards.

The expectation is that the greatest economic benefit will derive from using the same standards used by the civilian commercial community. This economic benefit is expected to derive in part from much greater flexibility with regard to opportunities to obtain upgraded capabilities at low cost by using COTS software. For the CCIS community this means that new technology can be added to existing systems faster and less expensively than having to contract for their addition to military-unique systems. To have any hope of wide-spread interoperability, standards are required at every interface: wherever data is transferred or connectivity is required. If open standards are adopted, there is a good chance that commercial products can be used in the CCIS. There are two risks associated with such an assumption: first that commercial product developers will adopt the standards and correctly adhere to them, second that such commercial products will be robust enough to be included in a secure military system. DISA can help to assure that its needs will be met by taking an active role in standards development. Since, however, there are so many relevant standards that participation in all would be prohibitive, selection of those most critical will

be needed. The use of such open standards leads to an open systems environment. However, there is no evidence that such a design approach will work in the defense environment. Developing such evidence is the basis for the following recommendation:

Recommendation. Establish a national CCIS test bed.

Some of the problems described thus far are based on lack of experience while others reflect a lack of detailed knowledge of the suitability of a specific standard or combination of standards for CCIS use. Those details should be explored in advance of making commitments to specific programs. Quantitative performance requirements will be needed in developing a specific system design. The ability of technology to meet these requirements will be an important part of the mission for the test bed.

Recommendation. Establish an ongoing process to upgrade the target profile and the generic CCIS architecture, especially with experience gained by implementors and users, feeding their ideas and problems back into the design process. Regular upgrades to adopt new technology and new requirements are needed, probably at intervals of two or three years.

Experience from the CCIS testbed and other advanced technology explorations must also be factored in. While standards, once approved, change only at infrequent intervals, technology continues to evolve and new standards are needed. The target profile will need to be updated periodically.

Recommendation. Support research on how to make policy and doctrine explicit within the system in order to provide flexibility.

As C² systems interoperate, they will have to accommodate differing doctrines and operational policies. If policy and doctrine were explicit in the CCIS, it would, for example, have a basis for detecting and accommodating inconsistencies.

Recommendation. Develop guidelines for application developers on how to make their applications portable.

This is an important step in achieving the desired results. The guidance should be considered a part of the architecture and evolve with its technical specifications.

Recommendation. Develop data definition standards for all CCIS mission areas and functions. It is recognized that much of this is currently underway.

This recommendation is included to reinforce the importance of that work by indicating that it is also needed to accomplish the goals of this architecture.

TABLE OF CONTENTS

1. INTRODUCTION TO THE ARCHITECTURE	1
1.1 ARCHITECTURE AND DESIGN	1
1.2 THE REQUIREMENT FOR OPEN SYSTEMS	3
1.3 STANDARDS AND PROFILES	5
1.4 DERIVATION OF THE TARGET PROFILE	6
1.5 DOCUMENT OVERVIEW	6
2. MILITARY INFORMATION SYSTEMS	7
2.1 NATIONAL MILITARY SECURITY OBJECTIVES AND STRATEGY	7
2.2 CCIS MISSION	9
2.3 CCIS FUNCTION	10
2.4 CCIS NEEDS	13
2.4.1 Interoperability	14
2.4.2 Survivability	17
2.4.3 Flexibility	18
2.4.4 Mobility	19
2.4.5 Affordability	20
2.4.6 Security	20
2.5 SUMMARY AND RELATIONSHIP TO SERVICE AREAS	22
3. THE GENERIC CCIS ARCHITECTURE	25
3.1 GOALS AND STRUCTURE	25
3.2 TECHNICAL IMPLICATIONS OF THE ARCHITECTURE	26
3.2.1 Interoperability	26
3.2.2 Survivability	28
3.2.3 Flexibility	28
3.2.4 Mobility	29
3.2.5 Affordability	29
3.3 OPERATIONAL POLICIES	30
3.4 THE USER'S VIEW OF THE CCIS	32
3.5 THE CCIS FUNCTIONAL ARCHITECTURE	33
3.5.1 Standard Services and Interfaces	36
3.5.2 Practical Aspects of the Architecture	39
3.5.3 Parallelism and Synchronicity	40
3.5.4 Cost Considerations of the Architecture	41
3.6 NETWORK SERVICES	42
3.6.1 Network Operation and Reconfigurability	45
3.7 OPERATING SYSTEM SERVICES	46
3.8 PROGRAMMING SERVICES	46

3.9 DATA MANAGEMENT SERVICES	47
3.10 SECURITY SERVICES	49
3.11 USER INTERFACE SERVICES	51
3.12 DATA EXCHANGE STANDARDS	53
3.13 DESIGN AND EVOLUTION OF THE CCIS	54
 4. WAM TARGET PROFILE: 1995-97	55
4.1 DATA EXCHANGE STANDARDS	55
4.1.1 Formatted Documents	56
4.1.2 Unformatted Documents	58
4.1.2.1 Office Document Architecture/Office Document Interchange Format (ODA/ODIF)—ISO 8613	58
4.1.2.2 Standard Generalized Markup Language (SGML)—ISO 8879	58
4.1.2.3 Document Style Segmentation and Specification Language (DSSSL)—ISO 10179	59
4.1.2.4 Standard Page Description Language (SPDL)	60
4.1.3 Computer Graphics Metafile (CGM)—ISO 8632	60
4.1.4 Standard for the Exchange of Product Model Data (STEP)—ISO 10303	60
4.1.5 Maps and Geographic Information Standards	61
4.1.5.1 Spatial Data Transfer Standard (SDTS)	61
4.1.5.2 Vector Product Standard (VPS)	62
4.1.5.3 Digital Geographic Information Exchange Standard (DIGEST)	62
4.1.6 Multimedia Standards	62
4.1.6.1 Multimedia Personal Computer (MPC)	62
4.1.7 Data Compression Standards	63
4.1.7.1 Joint Photographic Experts Group (JPEG)—ISO 10918	63
4.1.7.2 Motion Picture Experts Group (MPEG)—ISO 11172	64
4.1.7.3 Joint Bilevel Imaging Group (JBIG)	64
4.1.7.4 Video Coders for Audiovisual Services—CCITT H.261 and ANSI	64
4.2 DATA MANAGEMENT SERVICES	65
4.2.1 Reference Model of Data Management—ISO 10032	65
4.2.2 SQL-2—ISO 9075.2	66
4.2.3 Information Resources Dictionary System (IRDS)—ISO 10027	66
4.2.4 Remote Database Access (RDA)—ISO 9579	67
4.2.5 Transaction Processing (TP)—ISO 10026	67
4.3 NETWORK SERVICES	67
4.3.1 ISO OSI Reference Model—ISO 7498	68
4.3.2 Government Open Systems Interconnection Profile (GOSIP)	69
4.3.3 Directory Services—CCITT X.500, ISO 9594	70
4.3.4 File Transfer, Access, and Management (FTAM)—ISO 8571	71
4.3.5 Message Handling System (MHS)—CCITT X.400	71
4.3.6 Network Management	71
4.3.7 Virtual Terminal (VT)—ISO 9040, 9041	72

4.3.8	Remote Procedure Call (RPC)	72
4.3.9	Telematic (Teletex, Textfax, Telefax) Standards	72
4.4	OPERATING SYSTEM SERVICES	73
4.4.1	Basic Operating System Services—IEEE Standard 1003.1-1990	75
4.4.2	Ada Bindings—IEEE P1003.5	75
4.4.3	Real-Time Extensions—IEEE P1003.4	76
4.4.4	Security Extensions—IEEE P1003.6	77
4.4.5	Shell and Utilities—IEEE P1003.2	77
4.4.6	System Administration—IEEE P1003.7	77
4.4.7	Distributed System Services	78
4.5	PROGRAMMING SERVICES	78
4.5.1	Ada—ANSI/MIL-STD-1815A, ISO 8652, and Ada 9X	79
4.5.2	C Programming Language—ANSI X3.159-1989	80
4.5.3	C++ Programming Language	80
4.5.4	Other Languages	80
4.5.5	CAIS-A—MIL-STD-1838A	81
4.5.6	ECMA PCTE+	81
4.6	SECURITY SERVICES	81
4.7	USER INTERFACE STANDARDS	87
4.7.1	User Interface Architecture	88
4.7.1.1	Window Manager Standardization	88
4.7.1.2	Toolkit and User Interface Management System (UIMS) Standardization	88
4.7.2	Graphics Services	89
4.7.2.1	Graphics Subroutine Library Standards	89
4.7.2.2	Device Independent/Device Dependent Standardization	90
4.7.2.3	Color Standard	91
4.7.2.4	Rendering Standardization	91
4.7.3	Human Factors Standards	92
5.	DISCUSSION	93
5.1	OPEN SYSTEMS AND THE GENERIC CCIS ARCHITECTURE	93
5.1.1	Observations	93
5.1.2	Recommendations	95
5.2	THE ARCHITECTURE MANAGEMENT PROCESS	96
5.2.1	Observations	96
5.2.2	Recommendations	97
5.3	THE RELATIONSHIP BETWEEN THE ARCHITECTURE AND APPLICATIONS	98
5.3.1	Observations	98
5.3.2	Recommendations	100
5.4	THE TARGET PROFILE	100
5.4.1	Data Management	100
5.4.2	User Interface	100

5.4.3 Network Services	101
5.4.4 Operating Systems	101
5.4.5 Security	101
5.4.6 Programming Services	102
5.4.7 Data Exchange	102
5.5 DISCUSSION OF ISSUES	103
5.5.1 Network Services	104
5.5.2 Operating Systems Services	104
5.5.3 Programming Services	104
5.5.4 Data Management Services	105
5.5.5 Security Services	106
5.5.6 User Interface Services	107
5.5.7 Data Exchange Standards	108
5.6 FINAL NOTE	108

APPENDIX A — DATA EXCHANGE STANDARDS	A-1
A.1 INTRODUCTION	A-1
A.1.1 Definitions	A-3
A.1.2 Related Standards	A-4
A.2 FORMATTED DOCUMENTS	A-4
A.2.1 CCIS Requirements	A-7
A.2.2 Related Technology	A-7
A.2.2.1 State of the Practice	A-7
A.2.2.2 State of the Art	A-8
A.2.2.3 Future Research	A-9
A.2.3 Standards	A-9
A.2.3.1 Current Standards	A-9
A.2.3.2 Current Standards Activities	A-11
A.2.3.3 Future Standards Activities	A-11
A.3 UNFORMATTED DOCUMENTS	A-12
A.3.1 CCIS Requirements	A-12
A.3.2 Related Technology	A-12
A.3.2.1 State of the Practice	A-12
A.3.2.2 State of the Art	A-12
A.3.2.3 Future Standards Activities	A-13
A.3.3 Standards	A-13
A.3.3.1 Current Standards	A-13
A.3.3.2 Current Standards Activities	A-21
A.3.3.3 Future Standards Activities	A-22
A.4 DATA EXCHANGE FOR GRAPHICAL DATA	A-23
A.4.1 CCIS Requirements	A-23
A.4.2 Related Technology	A-23
A.4.2.1 State of the Practice	A-23
A.4.2.2 State of the Art	A-24

A.4.2.3 Future Research	A-24
A.4.3 Standards	A-24
A.4.3.1 Current Standards	A-24
A.4.3.2 Current Standards Activities	A-27
A.4.3.3 Future Standards Activities	A-28
A.5 MAPS AND GEOGRAPHIC INFORMATION	A-28
A.5.1 CCIS Requirements	A-29
A.5.2 Related Technology	A-29
A.5.2.1 State of the Practice	A-29
A.5.2.2 State of the Art	A-30
A.5.3 Standards	A-31
A.5.3.1 Current Standards	A-31
A.5.3.2 Current Standards Activities	A-31
A.6 METEOROLOGICAL DATA	A-33
A.6.1 CCIS Requirements	A-33
A.6.2 Related Technology	A-33
A.6.2.1 State of the Practice	A-33
A.6.2.2 State of the Art	A-33
A.6.2.3 Future Research	A-34
A.7 VIDEO	A-34
A.7.1 CCIS Requirements	A-34
A.7.2 Related Technology	A-35
A.7.2.1 State of the Practice	A-35
A.7.2.2 State of the Art	A-35
A.7.3 Standards	A-37
A.7.3.1 Current Standards	A-37
A.7.3.2 Current Standards Activities	A-38
A.8 AUDIO	A-38
A.8.1 CCIS Requirements	A-38
A.8.2 Related Technology	A-39
A.8.2.1 State of the Practice	A-39
A.8.2.2 State of the Art	A-39
A.8.3 Standards	A-40
A.8.3.1 Current Standards	A-40
A.8.3.2 Current Standards Activities	A-40
A.9 DATA COMPRESSION	A-40
A.9.1 CCIS Requirements	A-40
A.9.2 Related Technology	A-41
A.9.2.1 State of the Practice	A-41
A.9.2.2 State of the Art	A-41
A.9.2.3 Future Research	A-41
A.9.3 Standards	A-42
A.9.3.1 Current Standards	A-42

A.9.3.2 Current Standards Activities	A-42
A.10 ISSUES	A-44
APPENDIX B — DATA MANAGEMENT	B-1
B.1 INTRODUCTION	B-1
B.2 CCIS DATA MANAGEMENT NEEDS	B-2
B.3 HARDWARE AND SOFTWARE TECHNOLOGY ASSESSMENT	B-3
B.3.1 Near-term Hardware Technologies	B-3
B.3.1.1 General Hardware Improvements	B-3
B.3.1.2 Data Management Specific Hardware Improvements	B-5
B.3.2 Near-term Software Technologies	B-6
B.3.2.1 General Software Improvements	B-6
B.3.2.2 Data Management Specific Software Improvements	B-7
B.3.3 Long-Term Hardware Technologies	B-9
B.3.4 Long-Term Software Technologies	B-9
B.4 STANDARDS ASSESSMENT	B-10
B.4.1 Reference Model of Data Management	B-10
B.4.2 Information Resource Dictionary System	B-12
B.4.2.1 IRDS Standardization	B-12
B.4.2.2 Outlook	B-13
B.4.3 Database Language SQL	B-14
B.4.3.1 SQL Standardization	B-14
B.4.3.2 Interface Considerations	B-14
B.4.3.3 FIPS PUB 127-1	B-15
B.4.3.3.1 Provisions	B-15
B.4.3.3.2 Procurement Considerations	B-15
B.4.3.3.3 FIPS Flagger	B-16
B.4.3.3.4 NIST Conformance Testing	B-17
B.4.3.4 Current SQL Standardization Activities	B-17
B.4.3.5 Outlook	B-18
B.4.4 Remote Database Access	B-19
B.4.5 POSIX Database Services	B-19
B.4.6 Other Data Management Related Efforts	B-20
B.5 ISSUES	B-20
APPENDIX C — NETWORK SERVICES	C-1
C.1 INTRODUCTION	C-1
C.2 THE OSI/GOSIP LAYERED MODEL	C-3
C.3 APPLICATION SERVICES	C-7
C.3.1 File Transfer, Access, and Management (FTAM)	C-7
C.3.2 Telematic Services	C-8
C.3.3 Message Handling System	C-9
C.3.3.1 Virtual Terminal	C-10
C.3.4 Distributed Transaction Processing	C-11

C.3.5 Job Transfer and Manipulation	C-12
C.3.5.1 OSI Jobs	C-12
C.3.6 Remote Database Access	C-14
C.3.7 Network Management	C-15
C.3.8 Directory Services	C-17
C.3.9 Interactive Graphics	C-17
C.3.10 Teleconferencing	C-18
C.4 UPPER LAYER SERVICES	C-18
C.4.1 Application Layer Services	C-18
C.4.2 Presentation Layer Services	C-20
C.4.3 Session Layer Services	C-20
C.5 TRANSPORT LAYER SERVICES	C-21
C.6 LOWER LAYER SERVICES	C-22
C.6.1 X.25 WAN	C-22
C.6.2 Connectionless Network Service	C-23
C.6.3 LAN Link Control	C-24
C.6.4 Fiber Distributed Digital Interface	C-25
C.6.5 ISDN	C-28
C.6.6 BISDN	C-30
C.6.7 LAN	C-30
C.7 NETWORKING ISSUES	C-32
C.7.1 Security	C-32
C.7.2 Performance	C-33
 APPENDIX D — OPERATING SYSTEMS SERVICES	 D-1
D.1 INTRODUCTION	D-1
D.1.1 Scope	D-1
D.1.2 Background	D-2
D.1.2.1 Open Systems and POSIX	D-2
D.1.2.2 Generic CCIS Architecture	D-5
D.1.2.3 Operating System Services	D-8
D.2 WAM REQUIREMENTS	D-9
D.2.1 Interactive, Multi-Tasking, Multi-User Processing	D-10
D.2.2 Distributed Processing	D-11
D.2.3 Fault Tolerance	D-12
D.2.4 Real Time	D-12
D.2.5 Security	D-13
D.2.6 System Administration	D-14
D.2.6.1 Ada	D-14
D.3 DERIVED OPERATING SYSTEM REQUIREMENTS	D-15
D.3.1 Interactive, Multi-Tasking, Multi-User Processing	D-15
D.3.1.1 Operating System Support for Interactive, Multi-Tasking, Multi-User Processing	D-16
D.3.1.2 POSIX Support for Interactive, Multi-Tasking, Multi-User	

Systems	D-19
D.3.2 Distributed Processing	D-21
D.3.2.1 Operating System Support for Distributed Computer Systems ..	D-21
D.3.2.2 POSIX Support for Distributed Computer Systems	D-22
D.3.3 Real Time	D-25
D.3.3.1 Operating System Support for Real-Time Computing	D-26
D.3.3.2 POSIX Support for Real-Time Computing	D-27
D.3.4 Security	D-29
D.3.4.1 Operating System Support for Security	D-29
D.3.4.2 POSIX Support for Security	D-31
D.3.5 System Administration	D-31
D.3.5.1 Operating System Support for System Administration	D-32
D.3.5.2 POSIX Support for System Administration	D-33
D.3.6 Ada	D-33
D.3.6.1 Operating System Support for Ada	D-33
D.3.6.2 POSIX Support for Ada	D-35
D.4 CONCLUSIONS	D-36
D.4.1 Analysis Of Derived Requirements	D-36
D.4.2 Overall Evaluation Of POSIX Support Of Generic CCIS Requirements	D-37
D.4.2.1 Anticipated Benefits	D-38
D.4.2.2 Potential Risks	D-38
D.4.2.3 Recommendations for Minimizing Risks	D-39
APPENDIX E — PROGRAMMING SERVICES	E-1
E.1 INTRODUCTION	E-1
E.2 SOFTWARE DEVELOPMENT ENVIRONMENT	E-2
E.2.1 General Requirements and Views	E-2
E.2.1.1 Software Developer/Integrator View	E-4
E.2.1.2 End-User View	E-5
E.2.2 Languages	E-6
E.2.2.1 Current Standards	E-9
E.2.2.2 Languages for a Future CCIS	E-10
E.2.2.3 Ada 9X	E-10
E.2.2.4 Knowledge Representation Languages	E-10
E.2.2.5 Other Languages	E-10
E.2.3 Environments	E-11
E.2.3.1 Current Standards	E-11
E.2.4 Tools	E-13
E.2.4.1 Current Standards	E-15
E.2.4.2 Tools for a Future CCIS	E-15
E.2.4.3 Expert System Tools	E-16
E.2.5 Process Model and Development Method	E-17
E.2.5.1 Current Standards	E-19

E.2.6 Reuse Support	E-19
E.2.6.1 Reuse Library Issues	E-20
E.2.6.2 Current Standards	E-21
E.3 OPERATIONAL ENVIRONMENT	E-21
E.3.1 General Description	E-21
E.3.2 Ada View	E-22
E.3.2.1 Operating System Services	E-24
E.3.2.2 Ada Runtime Services	E-25
E.3.3 Views of an Operational Environment	E-28
E.4 ARCHITECTURAL ISSUES	E-31
E.4.1 Single vs. Dual Environment Implementation	E-31
E.4.2 Environment Support	E-32
E.4.3 POSIX Services for Ada	E-33
E.5 SUMMARY	E-34
APPENDIX F — SECURITY SERVICES	F-1
APPENDIX G — USER INTERFACE SERVICES	G-1
G.1 INTRODUCTION	G-1
G.2 HUMAN-COMPUTER INTERACTION	G-1
G.2.1 Basic Interaction Types	G-2
G.2.2 Direct Manipulation	G-4
G.2.3 Group Interfaces	G-5
G.3 USER INTERFACE ARCHITECTURE	G-6
G.3.1 Dialogue Layer	G-7
G.3.2 Presentation Layer	G-9
G.3.3 Toolkit Components and Subroutine Foundation Layers	G-10
G.3.4 Data Stream Interface Layer	G-11
G.3.5 Data Stream Encoding Layer	G-11
G.4 GRAPHICS SERVICES	G-12
G.4.1 Graphical Kernel System (GKS) and GKS-3D	G-12
G.4.2 Programmers Hierarchical Interactive Graphics System (PHIGS)	G-13
G.4.2.1 PHIGS Extension for X (PEX)	G-13
G.4.3 Device-independent/device-dependent (DI/DD) Graphics Standards ..	G-14
G.4.4 Rendering Standards	G-15
G.5 INPUT/OUTPUT DEVICES	G-15
G.5.1 Display Devices	G-15
G.5.1.1 CRT Displays	G-16
G.5.1.2 Flat Panel Displays	G-19
G.5.1.3 Projection Displays	G-20
G.5.1.4 3D Displays	G-21
G.5.2 Input Devices	G-22
G.5.2.1 Keyboard	G-22
G.5.2.2 Pointing Devices	G-23

G.5.2.3 Scanners and Digitizer Tablets	G-24
G.5.2.4 Touch Technologies	G-25
G.5.2.5 3D Input Devices	G-26
G.5.3 Video Devices	G-27
G.5.4 Hardcopy Devices	G-27
G.5.4.1 Printers	G-27
G.5.4.2 Plotters	G-28
G.6 GRAPHICS HARDWARE	G-30
G.6.1 Bitmaps and Rasterops	G-31
G.6.2 Colormaps	G-31
G.6.3 Graphics Accelerators	G-32
G.6.4 PC Graphics Boards	G-33
G.6.5 Graphics Workstations	G-33
G.7 SPECIAL APPLICATIONS	G-35
G.7.1 Visualization	G-35
G.7.2 Hypertext	G-37
G.7.3 Teleconferencing	G-39
G.7.4 Desktop Conferencing	G-40
G.7.5 Audio/Video Telecommunication	G-43
G.8 REQUIREMENTS FOR HUMAN FACTORS ENGINEERING	G-43
G.8.1 Applicable Documents	G-44
G.8.2 Workplace Environment	G-46
G.8.3 Workplace Layout	G-47
G.8.3.1 Adjustability and the Anthropometry of the User Population	G-48
G.8.3.2 Chair	G-48
G.8.3.3 Screen Position, Orientation, and Viewing Distance	G-49
G.8.4 Human Factors Engineering Principles	G-50
G.9 OUTSTANDING ISSUES	G-51
G.9.1 User Model	G-51
G.9.2 User Interface Development	G-52
G.9.3 Aids For Cognitive Processing	G-52
G.9.3.1 Interaction History Aids	G-53
APPENDIX H — WAM INFORMATION NEEDS	H-1
H.1 INTRODUCTION	H-1
H.2 BACKGROUND	H-1
H.2.1 WWMCCS Purpose and Scope	H-1
H.2.2 Evolution of WWMCCS	H-2
H.2.3 WWMCCS Today and Tomorrow	H-5
H.2.3.1 WWMCCS Today	H-5
H.2.4 What WWMCCS Must Support	H-8
H.2.4.1 Support to NCA/JCS	H-8
H.2.4.2 Support to CINCs	H-10
H.2.4.3 Support to Deliberate Planning and Crisis Management	H-11

H.2.5	Planned WWMCCS Improvements	H-14
H.3	WWMCCS REQUIREMENTS	H-16
H.3.1	General	H-16
H.3.2	Responsibility Level	H-17
H.3.3	Functional Needs	H-18
H.4	WAM TARGET ARCHITECTURE 1995-1997	H-21
H.4.1	Information Needs	H-21
H.4.2	Target Profile Requirements by Service Area	H-22
H.4.2.1	Data Exchange	H-23
H.4.2.2	Data Management	H-24
H.4.2.3	Network Services	H-26
H.4.2.4	Operating System Services	H-29
H.4.2.5	Programming Services	H-30
H.4.2.6	Security	H-31
H.4.2.7	User Interface	H-31
REFERENCES	References-1
ACRONYMS	Acronyms-1

**THIS
PAGE
IS
MISSING
IN
ORIGINAL
DOCUMENT**

LIST OF FIGURES

Figure ES-1. Services and Interface Connectivity	x
Figure 1-1. Design Sequence	1
Figure 2-1. National Security Objectives and CCIS Functions	8
Figure 2-2. A View of the C2 Process	11
Figure 2-3. Force Capabilities Mapped to Military Needs	22
Figure 2-4. Satisfaction of CCIS Needs by Service Areas	22
Figure 3-1. Command Center Local Area Network	30
Figure 3-2. Peer-to-peer Communication	32
Figure 3-3. The User's View of the CCIS	32
Figure 3-4. CCIS Functional Architecture.....	33
Figure 3-5. CCIS Functional Architecture with Applications	34
Figure 3-6. Services and Interface Connectivity	38
Figure 3-7. Examples of a System Monitoring Itself.....	40
Figure 3-8. Top Level Connection Architecture.....	42
Figure 3-9. Mid-level Communication Architecture	43
Figure 3-10. Open Systems Interconnection (OSI) Architecture.....	44
Figure 3-11. The Role of Programming Services (Ideal vs. Realistic).....	46
Figure 3-12. Multiple Data Models and Location Transparency.....	48
Figure 3-13. Security in Relational Database Access.....	50
Figure A-1. Data Interchange Structure.....	A-2
Figure B-1. A Federated DBMS and its Components.	B-8
Figure B-2. Generic Model of Data Management	B-11
Figure C-1. MHS and FTAM Layered Model.....	C-5
Figure C-2. Scope of FTAM Agreements	C-7
Figure C-3. JTM Agencies.....	C-12
Figure C-4. FDDI Architecture.....	C-25
Figure C-5. FDDI LAN	C-27
Figure D-1. Single System.....	D-5
Figure D-2. A Single System in a Network (The POSIX View)	D-7
Figure E-1. Software Development Environment	E-2
Figure E-2. The Software Developer/Integrator View	E-5
Figure E-3. The End-User View	E-5
Figure E-4. APIs in the Operational Environment	E-21
Figure E-5. Ada Application on a Bare Machine	E-28
Figure E-6. Ada Application on a Machine with an Operating System	E-29
Figure G-1. Modes of Worker Collaboration	G-41
Figure H-1. The Elements of WWMCCS.....	H-5
Figure H-2. Tactical Command and Control Systems.....	H-7
Figure H-3. Crisis Action Procedures vs. Deliberate Planning Procedures.....	H-12
Figure H-4. Steps in Crisis Action Procedures (JCS Pub 5-02.4)	H-18

**THIS
PAGE
IS
MISSING
IN
ORIGINAL
DOCUMENT**

LIST OF TABLES

Table ES-1. The Target Profile	xi
Table 2-1. Electronic Warfare Threats and Vulnerabilities	21
Table 3-1. Security in the OSI Model	49
Table 4-1. Data Exchange Standards	55
Table 4-2. Why Data Exchange Standards May Not Be Needed in 1995	55
Table 4-3. Data Management Standards	65
Table 4-4. Network Services	68
Table 4-5. Operating System Services	73
Table 4-6. Programming Services	79
Table 4-7. Security Services	82
Table 4-8. User Interface Standards	87
Table A-1. Related Standards	A-4
Table B-1. Microprocessor Technology Forecast	B-4
Table C-1. Layered Model of Communications	C-3
Table C-2. Communications Media Capabilities	C-33
Table D-1. Layers of Operating System Architectures	D-5
Table D-1. Requirements versus Operating System Functions	D-15
Table E-1. Operating System Services and Functions	E-24
Table E-2. Checks Performed on Ada Predefined Exceptions	E-26
Table H-1. Comparison Between Small Contingencies and Large Operations	H-8
Table H-2. Key Responsibilities of the JCS	H-9
Table H-3. Crisis Action vs. Deliberate Planning	H-12
Table H-4. Composition of Responsibility Level	H-17
Table H-5. Summary of Information Requirements for the Supported Theater Level (Level II), Part 1	H-34
Table H-6. Summary of Information Requirements for the Supported Theater Level (Level II), Part 1	H-36
Table H-7. Summary of Information Requirements for the Supporting Level (Level III), Part 1	H-38

1. INTRODUCTION TO THE ARCHITECTURE

1.1 ARCHITECTURE AND DESIGN

A command and control information system (CCIS) is an integrated complex of doctrine, procedures, organizational structure, personnel, equipment, facilities, and communications that together provide authorities at all levels of command with timely and adequate data to use in planning, directing, and controlling their activities [JCS PUB 1-02, 77]. It consists of a group of nodes connected by communication mechanisms collectively called a network. A node is where work gets done, either by a user connecting to system resources through a personal computer, workstation, or specialized piece of equipment or by an autonomously operating device that works with no direct user interaction.

This document defines an architecture to guide designers of future military command and control information systems over the next twenty years. It is general, covering a range of command and control applications and sizes of system in the near future and in the long term: a *generic architecture*. It treats only the top level of the design, not details of the applications that will operate in the CCIS. However, there is a specific target for a first implementation of this generic architecture, the upgrade to WWMCCS (the Worldwide Military Command and Control System) known as WAM, the WWMCCS Automated Data Processing Modernization program, planned for 1995-97. Its guidelines are summarized in a *target profile* of recommended standards and protocols. As with the creation of any large-scale system, design proceeds from needs to implementation. Figure 1-1 shows a sequence of steps in the design process, including positions of generic architecture and target profile. There is a single path from needs to the generic architecture—with opportunity to backtrack at points—followed by a dual path from generic architecture to implementation. The left-hand part of the path takes a specific target period and proceeds through a specific design, while the right-hand side is concerned with transition and validation.

Modern warfare places heavy demands on C² systems, forcing technology to be inserted at increasingly rapid rates to maintain technical superiority. Reconciling the requirement for new technology with logistics needs is leading the Military Services to fol-

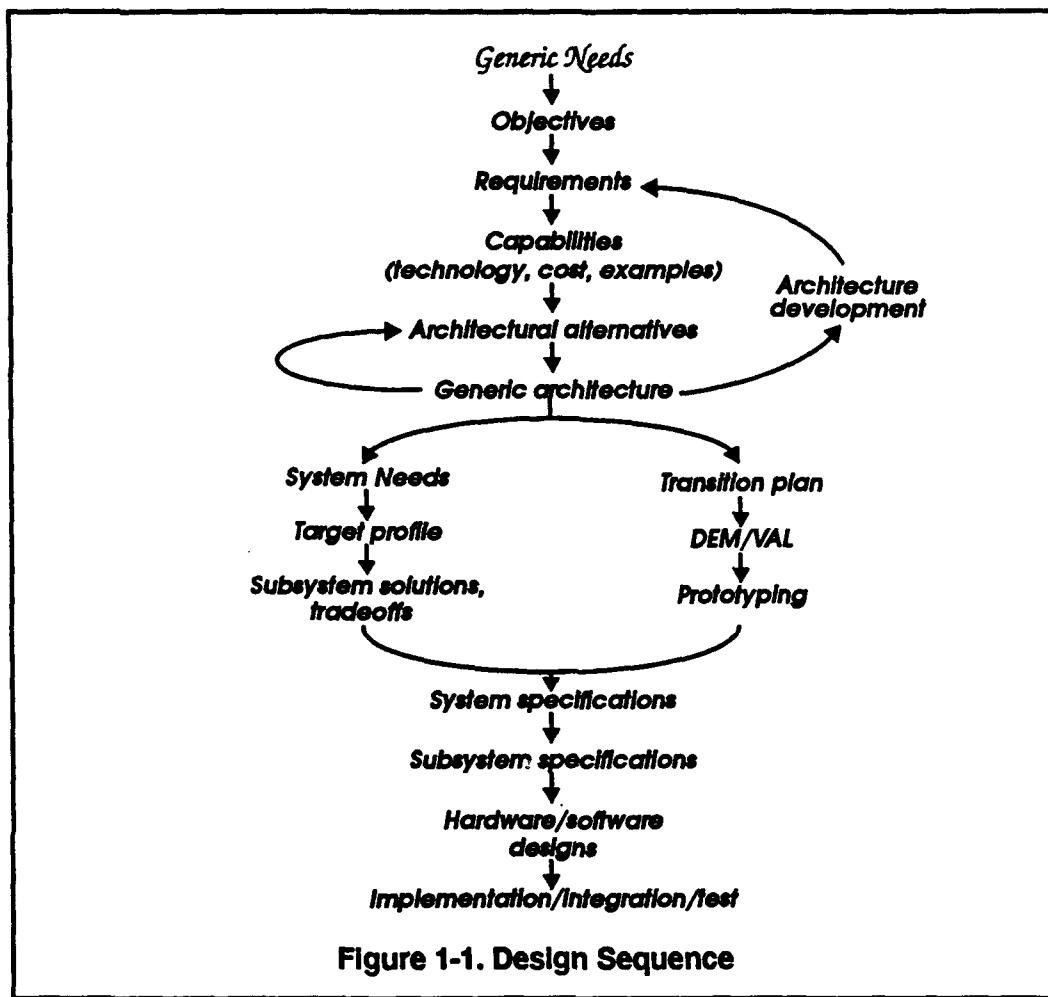


Figure 1-1. Design Sequence

low the Federal Government and industry in standardizing the interfaces between components rather than standardizing the components themselves. Most current military standards in the information systems domain apply to hardware. These standards do not appear in this document because its specification is above the level of hardware-software tradeoffs. Standards at the architectural level concern interfaces and interconnection protocols. Typical of such standards are the U. S. GOSIP (Government Open Systems Interconnection Profile) and POSIX (Portable Operating System Interface for Computer Environments) interface definitions discussed in detail in Sections 3 and 4. There is a trend toward applying national and international industry standards in defense systems. The Department of Defense's CIM (Corporate Information Management) initiative is assisting the process by centralizing the standards activities of the separate Services and coordinating them through NIST (the National Institute for Standards and Technology). Paul Strassmann, Director of Defense

Information, said in testimony to the Defense Subcommittee of the House Appropriations Committee on April 24, 1991,

We shall proceed without further delay to construct all DoD information systems according to approved Federal Standards as defined by the National Institute of Standards and Technology. We shall focus DoD resources on accelerated adoption of Federal Information Processing Standards (FIPS). We shall continue participating in international and industry standard organizations, after endorsement from the National Institute of Standards and Technology. . . All information standards activities in DoD shall be under central coordination from the new Center for Information Management within the Defense Communications Agency and guided by policy from the Director of Defense Information.

In summary, the generic CCIS architecture emphasizes interface and format standards—those that ease the transfer of information from site to site and from application to application. Interoperation with older equipment and foreign systems will benefit from use of standards, providing interface specifications where older systems adhere to standards. Where they do not, standards will at least provide a basis for conversion.

1.2 THE REQUIREMENT FOR OPEN SYSTEMS

The *WWMCCS ADP Modernization (WAM) Decision Coordinating Paper (DCP)* [WAM DCP 1989] is the motivating statement of requirements for the generic architecture and the target profile. It outlines a strategy for evolutionary improvement of WWMCCS that addresses application software and the underlying infrastructure, operating systems, database management systems, and network support. Emphasis is on incremental development and deployment of planning software and migration to an open system environment (OSE). The specifications that define an OSE are referred to as *open system standards*. As defined in the *Application Portability Profile (APP): The U. S. Government's Open System Environment Profile* [NIST APP/OSE 1991, 2], an open system environment (OSE) is based on an architectural framework that allows an extensible collection of capabilities. The APP is key to the generic architecture, serving as the basis for the breakdown of the architecture into service areas and providing a prototype of the profile approach to defining a future system. It is defined in terms of nonproprietary specifications available to any vendor and evolves through an open (public) consensus process. Several organizations, including the International Organization for Standardization (ISO), the American National Standards Institute (ANSI), the European Computer Manufacturers Association (ECMA), and the

Institute of Electrical and Electronics Engineers (IEEE), serve as public forums for the open consensus process of developing and evolving standards.

The roots of OSE lie in computer networking, where standard protocols and services are key to connectivity among heterogeneous computer systems. The OSE concept can be viewed as a generalization that goes beyond connectivity to information technology in general. It is heavily influenced by the ISO Open Systems Interconnection (OSI) reference model for computer networking (ISO 7498). In OSI, a seven-layer reference model serves as the architectural framework. The many networking protocol standards that have been developed in the context of the reference model serve as the *nonproprietary specifications*. An OSE has the potential for achieving important goals:

- a. Application portability. An OSE lays the foundation by specifying standard application program interfaces to standard services. Application software that uses the standard interfaces without proprietary enhancements is portable across different implementations of infrastructure software, the application-independent software that lies between the hardware and the application software and that provides the foundation on which the application software runs.
- b. System and application interoperability. OSE standards for networking, data management, and data exchange support the interoperation of heterogeneous computer systems and the application software running on them.
- c. Protection of software investments. Open system standards are designed to be independent of hardware technology. As technology evolves, new hardware platforms can implement the same standard interfaces as their predecessors and applications can be ported to the new platforms.
- d. Acquisitions from multiple sources. Since the standards are nonproprietary, different vendors can implement products that meet the standards and work together in a CCIS implementation.

The extent to which a specific OSE lives up to its potential depends upon the base of interest and support it achieves. That is, the success depends upon the number of customers demanding products conforming to the OSE, the number of vendors marketing products conforming to the OSE, and the number of organizations and individuals willing to support the consensus-based process for developing the standards underlying the OSE. Success also depends upon the farsightedness of standards developers. It is not always possible to design standard interfaces that can survive every revolutionary advance in technology. For

example, some current computer networking protocols will not support the high-speed data rates expected to become available in data communication networks later this decade. It is incumbent upon developers to consider scalability throughout the standardization process.

Although having an OSE is a given requirement, there is at present no hard evidence to support whether such an approach will be cost effective and whether it is suitable for defense systems security and reliability.

1.3 STANDARDS AND PROFILES

Two issues must be addressed in designing a particular OSE-based system: proliferation and generality of standards. The past two decades have seen the emergence of many computing standards, standards that compete with one another and that cannot work together in some cases. On the other hand, standards are meant to be as broadly applicable as possible, typically having many options, levels of compliance, and parameters. Specialization is required before conforming systems can be implemented. In its broadest sense, a profile is a suite of standards that are known to work together and that jointly meet the needs of an application domain. Profiles guide system designers in selecting standards.

GOSIP was developed by NIST to handle the transition towards OSI by the Federal Government, motivated by many competing and not quite compatible networking protocols. GOSIP specifies which protocols should be used at each layer. The APP is a “high-level” OSE profile aimed at Federal Government computing. It breaks its very broad domain of applicability into seven service areas and states which standards are to be used for each. POSIX, on the other hand, is very detailed, defining very narrow application domains. For example, one POSIX working group is defining profiles for four categories of real-time systems: minimal systems, controller systems, avionics systems, and multipurpose systems. Each of these profiles specifies not only standards, but also makes some options of the standards mandatory. A profile at this level precisely specifies an OSE and embodies design decisions tailored to the target application domain.

A profile is not an architecture. An architecture to specify the structure and a profile to specify the interfaces and protocols *combine* to provide direction to the systems designer.

1.4 DERIVATION OF THE TARGET PROFILE

The target profile identifies the standards to be used for a WAM system upgrade in the 1995-97 time frame. It was formulated through the following process:

- a. A set of requirements for a generic CCIS was compiled based on various open and classified documents and critical evaluations of the present WWMCCS.
- b. The NIST APP was adopted as a model for the target profile. Its framework of seven service areas: operating systems, user interface, programming services, data management, data interchange, graphics, and networks provides a good descriptive base. The target profile also has seven areas, but with two differences: security is separate and graphics has been merged into the user interface.
- c. A comprehensive survey of standards and standardization efforts in the seven service areas was conducted [Nash 1991]. The standards survey developed as part of the present task augments the standards in the NIST APP and gives a broader set to choose from in formulating the target profile.

There are several reasons for augmenting the APP. It is targeted at a broad spectrum of Federal computing, whereas the target profile is targeted to command and control. While the APP attempts to cover the demands of general-purpose computing, it does not necessarily cover the special demands of mission-critical computing and security. More importantly, the APP focuses on current needs and current standards, whereas the target profile is intended to cover 1995-97 and the generic architecture itself looks nearly two decades beyond. As a consequence of its time frame, the target profile includes emerging standards as well as industry and de facto pre-standard specifications.

1.5 DOCUMENT OVERVIEW

The remainder of the body of the document consists of four sections. Section 2 summarizes military C² requirements and shows how they relate to the generic CCIS. Section 3 describes the generic architecture. Section 4 covers the target profile. Finally Section 5 provides an evaluation of the architecture and discusses recommendations and issues arising from the definition. Appendices A-G delve into standards in more detail. Appendix H contains a summary of the requirements gleaned from various sources. References and acronyms appear at the end of the document.

2. MILITARY INFORMATION SYSTEMS

The problem of commanding and controlling armed forces and providing for effective information sharing is as old as war itself. Among warfare areas, C² is perhaps the most difficult to manage. Post-action reports of lessons learned frequently document shortfalls in information management. The problem of effective management and command is growing as a consequence of expanding intelligence, maneuver warfare, logistics, and administrative information processing needs. These demands on C² systems are further increased by the complexity of executing an integrated joint and allied military strategy. A command and control information system must provide the means to exercise command and control over U. S. forces in peacetime, crisis, and at all levels of warfare. It must have connectivity, flexibility, survivability, security, and interoperability sufficient to enable forces to be used to their maximum effectiveness operating together or with allies [DoD Defense 1988].

This section provides a review of the high-level requirements and general workings of a supporting CCIS. The basis of the CCIS architecture is explored by mapping overall national security objectives to desired CCIS function, characteristics, features, and service areas that satisfy specific planning, decision making, and execution requirements. The requirements have come from a variety of sources, all referenced as appropriate. A more detailed discussion of C² information systems requirements can be found in Appendix H.

2.1 NATIONAL MILITARY SECURITY OBJECTIVES AND STRATEGY

The general military features of a CCIS (summarized below in Section 2.4) depend on the overall strategy and the objectives that the system was designed to support. The current National Security Strategy, contained in the 1991 Joint Military Net Assessment and recently approved for submission to Congress by Admiral Jeremiah, Vice Chairman of the JCS [Jeremiah 1991], is broadly based and reflects a major role in the international environment. National security objectives include the following:

- a. Survival of the United States as a free and independent nation with its fundamental values intact and its institutions and people secure.

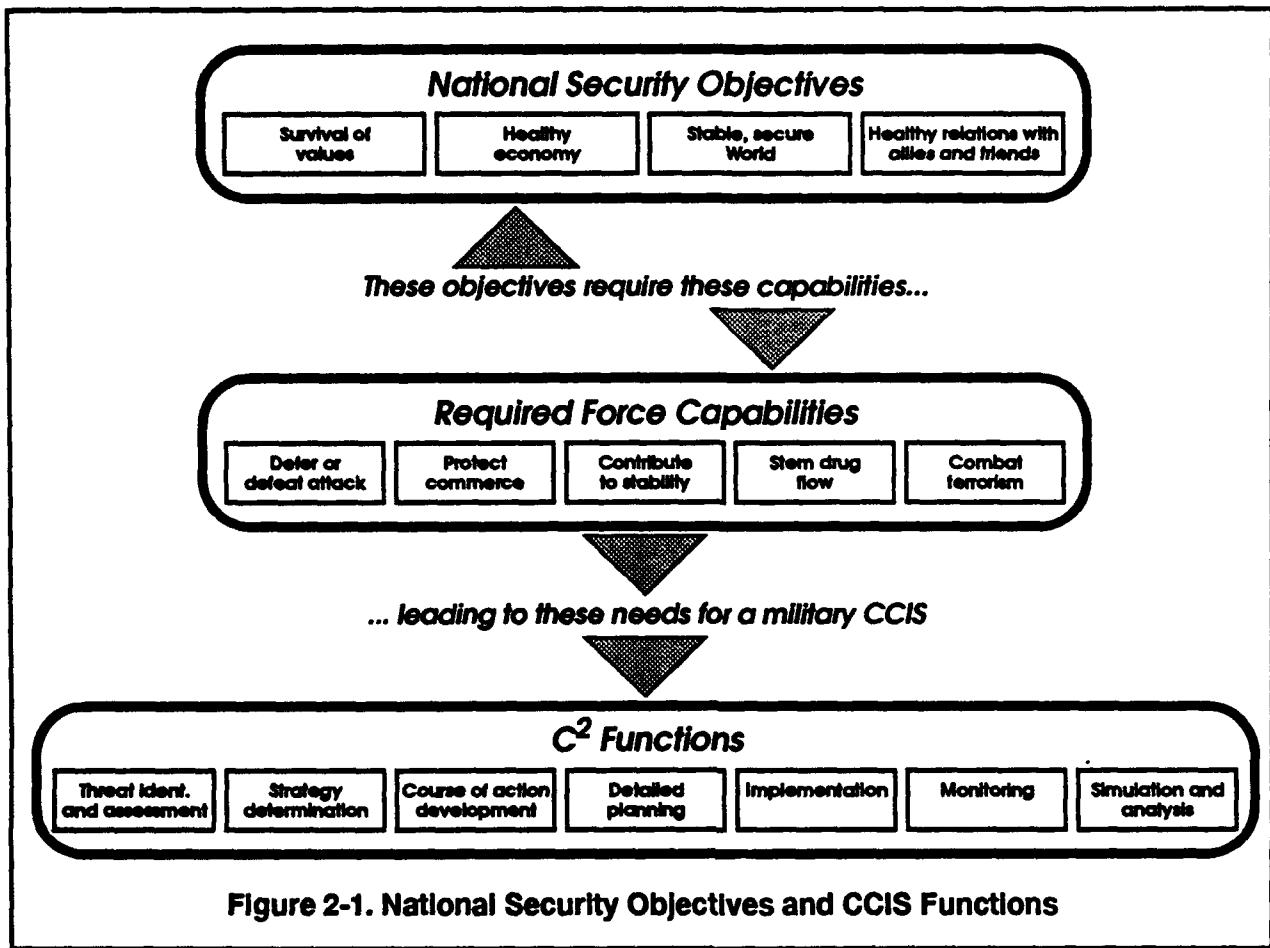
- b. A healthy and growing U. S. economy to ensure opportunity for individual prosperity and a resource base for national endeavors at home and abroad.
- c. A stable and secure world, fostering political freedom, human rights, and democratic institutions.
- d. Healthy, cooperative, and politically vigorous relations with allies and friendly nations.

The fundamental role of U. S. armed forces in supporting achieving the national objectives is to deter war, and, if this fails, to defend the nation's vital interest against any potential foe. The United States has global interests and responsibilities that require a force of wide-ranging capabilities to achieve national goals. The strategy is designed to perform the following:

- a. Deter military attack against the United States, its allies, and other countries whose interests are vital to our own, and to defeat such attack, singly or in concert with other nations, should deterrence fail.
- b. Protect free commerce; enhance the spread of democracy; guarantee U. S. access to world markets, associated critical resources, air and sea lines of communication, and space; and contribute to U. S. influence around the world.
- c. Contribute to regional stability through military presence, mutual security arrangements, and security assistance, and discourage, in concert with other instruments of power, policies and objectives inimical to U. S. security interests.
- d. Stem the production and transit of illegal drugs and their entry into the United States.
- e. Combat terrorism against U. S. citizens and help other national and international agencies combat it.

The relationship of these national military security objectives to required force capabilities is shown in Figure 2-1.

The strategy has broad implications for a military CCIS. The CCIS must be *global* in supporting U. S. world-wide interests: there is no single major adversary. The U. S. CCIS must *interoperate* with those of allies and other countries with similar interests. The strategy therefore requires a *mobile* and *flexible* CCIS, capable of use anywhere in the world and throughout the spectrum from low intensity conflict to nuclear warfare. Coalition or allied operations could involve conventional military operations. The CCIS may also sup-



port regional stability and low-level security assistance operations of training or "nation building." The CCIS must also support *counter-narcotics* and *counter-terrorism* operations, which require a *wide exchange of information* among DoD, civilian, and law enforcement agencies on national and international levels.

2.2 CCIS MISSION

The essential mission for a CCIS is to support a commander in exercising command and control of military forces. The DoD dictionary [JCS PUB 1-02] defines command and control as

The exercise of authority and direction by a properly designated command over assigned forces in the accomplishment of the mission. Command and control functions are performed through an arrangement of personnel,

equipment, communications, facilities, and procedures employed by a commander in planning, directing, coordinating, and controlling forces and operations in the accomplishment of the mission.

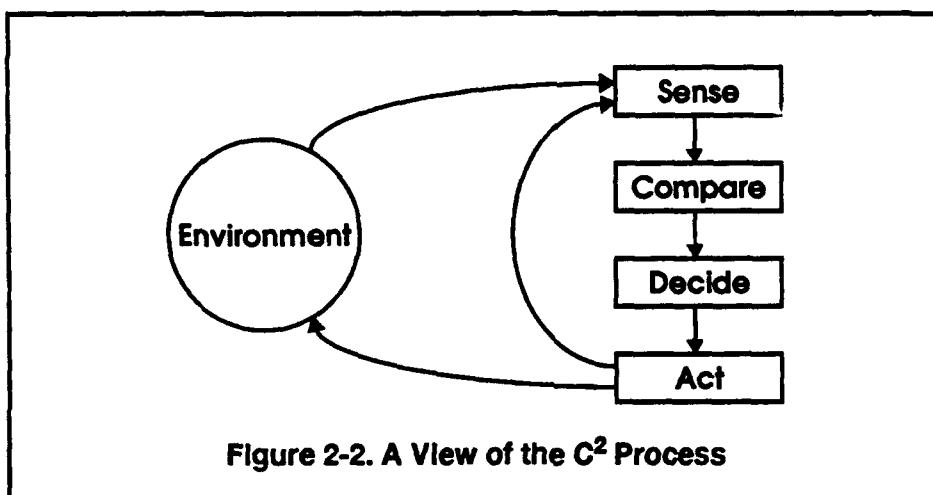
A CCIS supports the C² mission by providing the National Command Authorities (the NCA—the President and Commander-in-Chief and the Secretary of Defense or their duly deputized alternates or successors) the means to receive critical, time-sensitive warning and intelligence information on the military posture, readiness, and activities of friendly, neutral, and hostile forces. The NCA does this by applying the resources of the military departments, directing military operations, and providing guidance to the Unified and Specified Commands. A CCIS supports the C² systems of the Unified and Specified Commands on a non-interference basis. However, as operational and tactical information is generated within these commands, it is critical to the overall functioning of the CCIS and the informed support of the NCA for crisis planning, deployment, and execution that this information be integrated into the CCIS on a timely basis. As the needs are recognized for expanding tactical C² information exchange on the battlefield, the information exchange is being factored into doctrine operational planning. Increased amounts and kinds of data are also required to support increased automation of weapon and sensor systems. Weapons system automation, in turn, requires enhanced digital communications, e.g., for the targeting of precision guided weapons and reception of JSTARS (Joint Surveillance Target Attack Radar System) information. The long-term and expanding needs for information must be accommodated in all environments from peacetime to brink of war, to regional conflict, and, if required, through global nuclear war. A CCIS must support C² as long as military forces survive to fight.

While the focus of a military CCIS architecture is to provide control of assigned forces, the fundamental needs for data access, storage, processing, display, and transfer, with appropriate management controls, closely parallel other information system requirements. The automated information system which best supports the human decision-making process with timely information on status, situation assessment, operations planning, resource allocation, and execution will have a fundamental advantage over less capable systems.

2.3 CCIS FUNCTION

The function of a CCIS is to support an information flow for decision makers and responsible information recipients, such as commanders and staff officers. This information

must support strategic and tactical monitoring during peacetime, crisis, and wartime planning and execution activities. These CCIS activities are designed to complement the individual command and control decision process. One description of the process consists of observation, orientation, decision, and action (OODA). These steps form a decision loop. The decision maker who can complete the OODA loop faster and inside an opponent's decision cycle, by thinking more quickly and coherently, can not only react to events more rapidly but can control and shape the battle. There are related models such as that shown in Figure 2-2 providing a more detailed understanding of C². The figure illustrates the contin-



uous nature of the process as related to feedback occasioned by the commander's military decisions and the changing environment. The major players and decision-making components of the C² process are *human*. The CCIS that most efficiently supports C² through information acquisition, orienting displays, analytical and simulation assistance for comparisons and decision aids, and so forth, will provide the best use of assigned and limited resources by those humans.

Five planning and execution functions found in the Joint Operations Planning and Execution System (JOPES) are traditionally associated with a CCIS.

- a. *Threat Identification and Assessment.* Using observations from sensory and other processors to aid in identification and description of threats; provides the basis for strategy planning and resource allocation, developing suitable courses of action and plans, and subsequent monitoring and modification of planned actions during execution.

- b. *Strategy Determination.* Helping the operational commander formulate plans and initial courses of action; used to develop an optimal balance among threats, capabilities, and alternative strategies for global, multi-theater, regional, and individual crisis situations.
- c. *Courses of Action Development.* Testing of alternative actions after receipt of guidance and resource allocation decisions. Methods of testing feasibility and suitability of military options include considerations of readiness, mobilization, deployment, employment, and sustainment.
- d. *Detailed Planning.* Identifying actual forces, logistics, and optimum routing of forces to destination; supports development of the force, material and personnel lists, schedules, and supporting plans and directives to prepare an approved course of action.
- e. *Implementation.* Development of warning orders, alert orders, and commander's estimates and operations orders.

Two additional CCIS functions are needed to make the system work. They provide information and analytical support.

- f. *Monitoring.* The entire operation of the CCIS depends on timely information. Monitoring functions provide the medium by which users of the system obtain and use information relevant to each primary functional area, requiring gathering, storing, organizing, correlating, and communicating information. Monitoring is sometimes narrowly associated with sensor operations. However, in the larger sense, it applies to all sources of information with respect to the political-military situation, threat nature and capabilities, friendly active force readiness, allied force readiness, reserve component readiness, and material resource and sustainment status. Monitoring activities also include the incorporation of information received from situation reports (SITREPs), operational reports (OPREPs), and other messages.
- g. *Simulation and Analysis.* The analytical support provided for computer assisted or fully automated techniques in simulation and analysis of available CCIS information is crucial to the process. Peacetime, crisis, and wartime situation assessments must be made for current or anticipated threats, evaluation of enemy and allied capabilities, allocation of forces and resources, courses of action, and operation plans. Simulation must include deterministic and heuristic

models that deal with sustainment, transportation, movement for use in mobilization, deployment, and employment planning. They are also necessary to support wargaming analysis of operation plans. Finally, simulation and analysis must be afforded the standard statistical analysis tools, with graphics capabilities for presentation of outputs.

Support for the required force capabilities occurs by providing these core CCIS functions to commanders as user-in-the-loop decision makers.

2.4 CCIS NEEDS

An integrated planning, decision making, and execution system will provide a pool of capabilities that, combined with program applications, must satisfy military planning, decision making and execution features. Desired CCIS military requirements [JROC Mem 1989] may be summarized as:

- a. Enable national civilian and military leaders to propose, select, and implement preferred courses of action to achieve specific political and military objectives.
- b. Provide timely and accurate decision making tools that keep pace with crisis situations and support the detailed requirements of a deliberate planning process.
- c. Provide leaders and staff officers at all levels with timely, accurate, complete, and properly aggregated information that serves the decision-making process.
- d. Provide for continuous monitoring of the global and local situations to include the status of U. S. and friendly forces and resources, threat indications and warning, and the capabilities of potential adversaries.
- e. Provide analytical tools to support rapid development, evaluation, and selection of strategic options and military courses of action in single and multi-theater scenarios in concert with allies and law enforcement partners.
- f. Support execution planning requirements for development of operation plans (OPLANS) and operation orders (OPORDS) within specified time periods.
- g. Aid the evaluation of plans, including their underlying assumptions and probable implications, in order to assess capabilities, identify shortfalls, and decide resource priorities.

- h. Assist commanders to start, stop, or redirect military operations effectively in response to changes in guidance, resources, or threat.
- i. Aid mobilization, deployment, employment, and sustainment planning and execution.
- j. Integrate existing systems for planning, decision making, and execution within a single architecture defined by established standards and policies.
- k. Interface with existing and planned Service and Defense Agency, Allied, and other partner databases and information systems.
- l. Provide policies and procedures that are similar, if not identical, in peacetime, crisis situations, and war.
- m. Exploit technological advances in information systems and communications as required to maintain superiority over potential adversaries.
- n. Safeguard information from unauthorized access, manipulation, or retrieval and accidental or deliberate destruction.

The following sections describe characteristics of the CCIS architecture that will help achieve these needs.

2.4.1 Interoperability

Interoperability is defined in JCS PUB 1-02 as

- 1. The ability of systems, units, or forces to provide services to and accept services from other systems, units, or forces and to use the services so exchanged to enable them to operate effectively together.
- 2. The condition achieved among communications-electronics systems or items of communications-electronics equipment when information or services can be exchanged directly and satisfactorily between them [or] their users.

Interoperability among communications-electronics systems implies direct exchange of data or services. In basic digital information exchange, there must be a transmission path and the data must be interpreted at the different locations in a consistent manner, using previously agreed upon protocols, interfaces, and standards. Interoperability needs for digital information exchange among databases, security systems, cryptographic systems, and network management systems also exist. As warfare complexity grows, the required range of information and service exchange expands to include the following:

- a. Interoperation among all U. S. military CCISs at all echelons and including functional area systems.
- b. Interoperation with the CCISs of allies, again at all echelons and including functional area systems.
- c. Interoperation with directed civilian agency systems, such as those of law enforcement agencies.
- d. Interoperation in approved ways with information systems of the civilian community at large.

New joint warfare concepts mandate the need for integrated Army, Air Force, Marine Corps, and Navy Unified Command components; the U. S. military will also be expected to work closely with allies and non-DoD organizations. Interoperability will therefore become more important as diverse organizations need to work together. DoD policy [DoDD 4630.5] is very specific about these cases:

It is DoD policy to develop, acquire, and deploy tactical C³I systems and equipment that effectively meet the essential operational needs of the U. S. tactical forces, and that are compatible and interoperable where required with other U. S. tactical C³I systems and equipment, with Allied tactical C³I systems and equipment, and with U. S. non-tactical C³I systems and equipment. The degree of necessary interoperability shall be determined during the requirements validation process and shall be ensured through the acquisition process, deployment, and operational life of the system or equipment.

C³I (command, control, communications, and intelligence) has not been mentioned previously. It adds several layers of hardware through application function to the C² model. Command and control is still the base and the policy statement applies equally to C².

Uncontrolled proliferation of databases and applications across the armed forces has, as in civilian organizations, created a situation where few systems can accept data from each other. In a military CCIS, system components and applications must be able to communicate and share data without conversion delays or the expense of creating specialized conversion hardware or software. The required level of interoperation of the new systems with present WWMCCS installations remains to be defined, but the introduction of an open system environment based on common standards, shared applications, and more compatible software languages and operating systems will help significantly.

The Unified Command structure created after World War II was in part an effort to integrate mutual Service support within joint force organizations. Warfare complexity since then has continued to grow, and recent events in the Persian Gulf have highlighted the

importance of integrated battle management. This need goes well beyond the mission planning, coordination, and tasking authority of a joint force commander. The commander must be given the means to coordinate near real time convergence of joint forces in time and space. The joint task force (JTF) commander will have difficulty optimizing force employment if the Service component C² systems are not fully interoperable.

Another aspect of interoperability is the need for timely national intelligence support for targeting. National sensors and intelligence frequently provide detailed information outside the coverage of joint force sensors, e.g., target details or new satellite maps of the local terrain. Fusion, correlation, analytical assessment, and transmission of nationally generated target information to the operations planning cycle of a joint force is important to integrated battlefield targeting. Joint over-the-horizon-targeting using multi-Service sensors and sources is intended to provide an accurate picture of the over-the-horizon battlefield to support real-time targeting of long-range weapons.

If present trends continue, there may eventually be a single unified CCIS providing all functions and serving all echelons. Such unification should be encouraged but is unlikely in the foreseeable future. The ability of the United States to operate as part of an allied force in coalition warfare is increasingly important, politically and militarily. The Desert Storm operation pointed out the need for CCIS interoperability among allied partners not included in earlier interoperability planning. Combined force information transfer requirements for this operation ranged from character message exchange to processor-to-processor automated links. These interfaces supported mission planning, intelligence operations, targeting, air tasking order (ATO) assignments and exchange of liaison information. Such worldwide and unforeseen commitments require a CCIS to be sufficiently flexible that command and control information exchange (C²IE) nodes and automated interfaces can be placed where needed. However, unless there is an automated and interoperable interface between the U. S. and Allied C² systems, the flow of information will be restricted. The United States also has permanent allied force commitments to NATO and Korea where improved interoperability is a continuing goal.

Joint and Allied interoperability will be enabled by common data definitions, communication protocols, and crypto buffers. Interoperability between CCIS and communications systems will depend on national implementation of digital data transmission standards. However, the degree of interoperability achievable within the supported unified command is not entirely clear. Mission execution in most instances is accomplished by a JTF organized for that specific purpose. Consequently, it is assumed, as discussed in DoDD 5100.30,

that the command and control support systems of a JTF would be part of and interoperable with those in the unified commands.

Interoperability with civilian information systems is also a growing requirement as, for example, in the role of U. S. armed forces in stemming the production and transit of illegal drugs and working with other national and international agencies to combat terrorism. The ability to send and receive electronic mail (e-mail) is a major first step in interoperability between civilian and military C² information systems. Data file transfer and eventually direct intercommunication are on the horizon.

2.4.2 Survivability

A CCIS must survive as long as there are forces to control. The importance of anti-CCIS operations is reflected in Soviet radio electronic combat (REC) strategic and tactical doctrine which is directed at degrading those operations. Additionally, the Soviets have special force *Spetsnaz* teams trained for both physical and electronic attack of command centers. The United States also places considerable importance on C² warfare, reflected in a joint command, control, and communications countermeasures (C³CM) doctrine stressing the integrated use of operations security, military deception, jamming, and physical destruction supported by intelligence to deny information to, influence, degrade, or destroy adversary C³ capabilities. The object of both Soviet REC and U. S. C³CM doctrine is to deny adversary commanders and other decision makers the ability to operate their forces effectively. Uncovered CCIS locations will be attacked as a matter of priority, using all available physical and electronic means.

CCIS node survivability is expensive and difficult to achieve. There are few options in case of nuclear attack. Even in conventional conflicts, continuing normal C² operations when subject to physical or electronic attack is difficult. The keys to CCIS survivability are reduced observability, mobility, and redundancy. In every military operation, there are designated both a primary and an alternate command center. The availability of CCIS nodes, which provide redundancy and avoid targeting through mobility and use of readily movable surface or airborne platforms, is important to the success of an operation. Redundancy of databases is also important. Critical data must be stored and maintained at several locations.

Survivability of the military CCIS has a direct bearing on the survival of the United States as a free and independent nation. CCIS survivability can be enhanced by flexibility

of operations with multiple access points as well as relocation and reconstitution procedures designed to achieve primary, secondary, and emergency access pathways.

2.4.3 Flexibility

CCIS flexibility is the capability to adapt to changes in environment, force organization, structure, threat, employment doctrine, operational level, and new technology. Adaptability to new technology is sometimes termed *evolvability*. A CCIS should be equally capable in air, land, and maritime environments. Flexibility is a function of physical size, electrical power requirements, and robustness against environmental extremes. In a tactical deployment, size is important to aircraft or shipboard installation. On land, reduced size improves mobility and the capability to move a CCIS as far forward as a tactical situation might permit. Space is a rapidly expanding warfare environment. Tasking and control authority with the means to effect direct space platform control is, in certain situations, a highly desirable CCIS characteristic.

The CCIS should also be flexible in organization size and structure. Force sizes depend on the specific contingency and availability of strategic lift. The functional capabilities of a CCIS to support a commander must not vary as a function of size. Even a reduced size CCIS must be capable of performing all its functions at a level commensurate with the forces to be controlled. It must be capable of growing to support increased information needs as new or replacement forces and information gathering resources arrive in the area. It must also be capable of meeting force structure changes and considerably varied joint force organizations as, for example, when JTF structure evolves as a function of mission execution.

The loss of primary and first alternate CCIS nodes must be provided for in system design and network access. For example, the primary and designated alternate command centers (Division Command Groups A and B) could both be lost. This would result in the assumption of command by the next senior brigade or regimental command. At this point, the CCIS of the new commander at a lower tactical level would need a significantly enlarged span of control, which depends on the ability of the alternate CCIS immediately to expand.

Current land warfare doctrine emphasizes the importance of deep attack and maneuver warfare concepts. This doctrine stresses highly maneuverable surface and air mobile forces, and is designed to exploit opportunities against enemy vulnerabilities. Maneuver warfare

goals seek to circumvent force inequities, avoid attrition, and attack from positions of advantage. Relying on speed and surprise, strength is applied against enemy weakness. Success depends more on military competence than sheer superiority of numbers in troops and equipment. This concept places a premium on mission type orders to permit the tactical commander closest to the scene maximum latitude and freedom. The U. S. Army's Airland Battle Doctrine incorporates the maneuver warfare that was reflected in the Desert Storm 100-hour ground campaign.

Flexibility of the CCIS is also needed to adapt and evolve as new technology and capabilities are developed. The system will need to provide new and improved applications that expedite information exchange and to meet unique organizational situations or command preferences at certain CCIS nodes. System design must accommodate both new subsystem integration and the modification of applications. The introduction of an entirely new CCIS requires training user and maintenance personnel. Therefore new designs should be based on adapting to the current system as opposed to a "swap out" of an entire system, ensuring that training requirements can be satisfied with on-the-job training, providing the user operator maximum freedom to adapt the CCIS software to unique user needs, and ensuring the flexibility to introduce upgrades incrementally.

2.4.4 Mobility

Mobility is the ability of the CCIS to support nodes that can move independently. It must also support the ability of a node to disconnect from the network and to reconnect (with suitable validation and security precautions) at a different location. Success in achieving mobility will determine in large part whether forces under the control of commanders using the CCIS can achieve their full combat potential. It supplements redundancy in assuring CCIS survivability. A CCIS for a corps-size, highly mobile crisis reaction force will also require strategic mobility. Not only will a CCIS need to be deployed in minutes, it will also require (during deployment) connectivity with widely dispersed supporting forces to complete operations planning and execution.

A mobile CCIS is critical to a JTF and maneuver warfare doctrine. It enables seizing the military initiative and causing a hostile force to remain in reaction mode. The CCIS must be able to operate in forward areas accompanying tactical forces on the move, and be positioned sufficiently close to the source of tactical battle information that battlefield orientation and monitoring can be sustained. The quality and timeliness of directly observed

and accurately reported battlefield information throughout the entire CCIS is, in large part, based on mobility.

2.4.5 Affordability

The growth and costs of command and control systems since World War II has been phenomenal. In earlier years, C² cost was incidental. However, today's CCISs are involved in every aspect of command and control and are a major portion of the defense budget. The primary issue is how much automation of a CCIS is necessary or appropriate. This uncertainty is reflected in the questions raised by Creveld [1985, 3], "What are the strong points of man and which are those of machines? How should communication ('interfaces') between man and machine, as well as among the machines themselves, be organized?" Automation for its own sake must be avoided. Commercial developers create and market information systems that are basically satisfying, while military operational requirements frequently stretch the available technology often unnecessarily and expensively.

Affordability is a key CCIS characteristic and important to any CCIS architecture. The determination of how much automation is sufficient for "good enough" operational capabilities will always be a tough decision. As the relative cost of C² systems continues to mount, there is the distinct possibility that a planned CCIS could be unaffordable. However the availability of commercial off-the-shelf (COTS) products and non-development items (NDI) within a growing environment of common standards provides CCIS designers an opportunity to reduce overall purchase and long-term O&M (operation and maintenance) costs. There is much in common between a CCIS and a commercial information system, and there are many COTS products that could be applied to CCIS functions. Furthermore, the commercial software industry is rapidly moving towards open system concepts that will provide even greater opportunity. The continued development of commercial and military standards will also work to make both COTS and NDI more available and affordable.

2.4.6 Security

Security, an essential CCIS characteristic, has been subjectively measured in the past. It reflects the totality of protection and integrity features embedded in all the CCIS services and functions and includes a trustworthiness requirement that is as difficult as any other CCIS need. Such requirements such as multilevel security (MLS) may never be entirely

implemented, since solutions must be continuously improved to meet an ever-escalating threat. Electronic threats to the security of a CCIS are listed in Table 2-1. Passive threats

Table 2-1. Electronic Warfare Threats and Vulnerabilities	
Passive Electronic Surveillance Measures (ESM)	
Traffic analysis	
Eavesdropping	
Emanation monitoring	
Active Electronic Counter Measures (ECM)	
Wiretapping	
Intrusion	
Jamming	
Interference	
Software "viruses, worms" etc.	

may be most serious when dealing with the development of useful intelligence that permits hostile force exploitation. On the other hand, active forms of electronic interference, such as AM continuous wave, analog, FM noise, as well as chaff under certain conditions, might be difficult to overcome and could cause disruption. The increased use of software in the CCIS has expanded system vulnerability to internal and external data tampering, message stream modification, and malicious insertion of logic bombs, viruses, and worms.

For reasons of reliability and integrity, data should be collected and maintained close to the source (with appropriate redundancy and backup, of course). Frequently information comes from unclassified sources where the maintenance of a top-secret terminal is not practical. This is a major problem for reporting on mobilization readiness of reserve component units and coordinating their activation, initial movement, and eventual deployment. A small reserve unit cannot afford or maintain the facility and encryption requirements for top secret system-high operation. There is a similar problem for small unit tactical CCIS nodes that need to interface with tactical data systems in order to expedite reported information upwards within the CCIS. At these forward echelons, the operating level of security is much lower than at higher echelons.

A total solution to multilevel security may not be immediately forthcoming. However, alternative approaches based on low-risk technology and separation of functional compo-

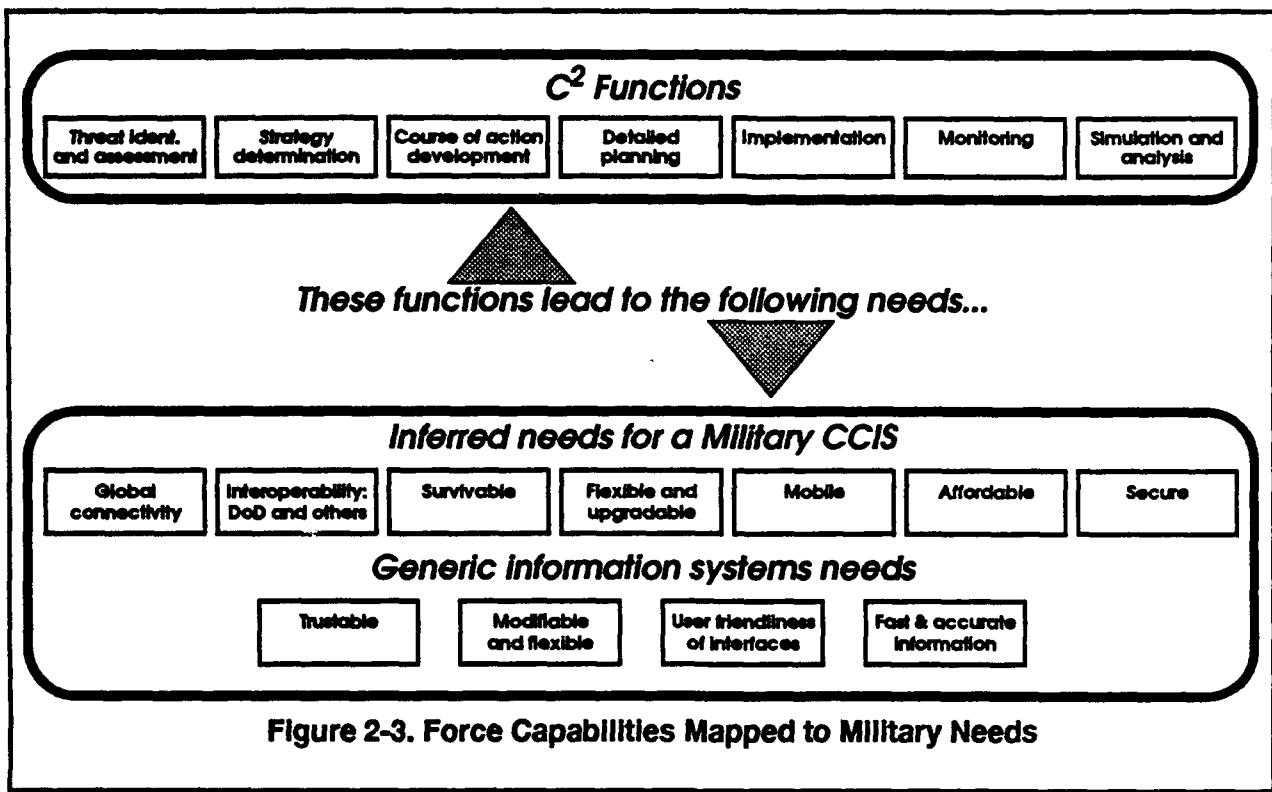
nents will provide adequate interim security. Subsequently, transition to full multilevel security can be made when development, certification, and accreditation are completed.

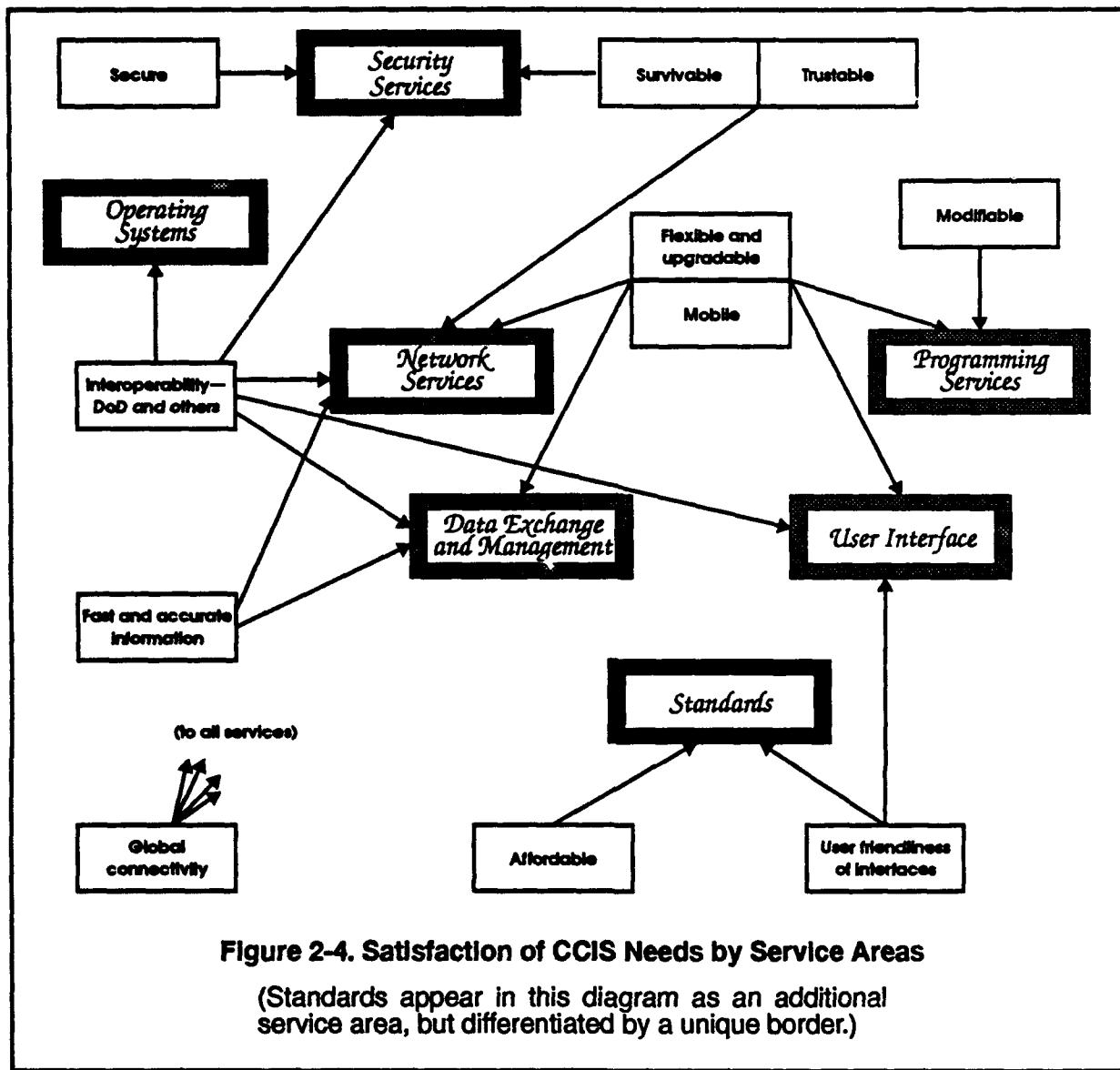
2.5 SUMMARY AND RELATIONSHIP TO SERVICE AREAS

The functionality of the generic CCIS architecture is based on seven service areas. These will be described in full detail in Sections 3 and 4, but in brief, they are as follows:

- a. Data exchange standards permit exchange of data among applications and systems in a way that preserves meaning and relationships.
- b. Data management services store, control, distribute, manage, and allocate information.
- c. Network services support logical and physical communication.
- d. Operating systems services manage hardware and software resources and interfaces controlling the movement of data and its interaction.
- e. Programming services support development and execution of applications.
- f. Security services provide secrecy, privacy, and integrity of programs and data and ensure availability to legitimate requesters.
- g. User interface services support visual and functional interaction.

The architectural service areas can be related to the overall architecture by 1) summarizing the national security objectives, 2) considering how required force capabilities meet these national security objectives, and 3) showing how the required force capabilities map onto the needs for the military CCIS and the prime CCIS service areas. The earliest figure, Figure 2-1, indirectly shows how national security objectives and required force capabilities relate to CCIS functions. These desired CCIS functions, as shown in Figure 2-3, when translated into objectives, provide a basis for understanding the inferred needs of a military CCIS. The CCIS needs reflected at the bottom of the figure support the needs of a generic information management system. Finally, the needs of both a military CCIS and an information system can be satisfied within the seven service areas as indicated by Figure 2-4.





3. THE GENERIC CCIS ARCHITECTURE

3.1 GOALS AND STRUCTURE

The military CCIS whose underlying structure is described in this section is the computer hardware and software that will, in the future, aid command and support personnel working to achieve a C² mission. The single overriding goal for this architecture is to enable the designer of a future CCIS to define a specific system that will achieve the C² mission by satisfying the requirements summarized in Section 2. The architecture describes the functional parts of a CCIS and how they interconnect. From the user's point of view the architecture provides a generic application program interface oriented towards C² applications. From the point of view of the services that the architecture will provide, it is a set of standards and protocols that assure the smooth, correct interaction of the various system components.

The architecture is referred to as a *functional architecture* to make clear the distinction between this description and a "systems architecture," a term conventionally used to describe a design in considerably more detail. It is not a system design and does not attempt to describe the hardware that makes up the nodes. Nor does it specify any geometry (physical layout) or technology used to configure the network or the applications that will run in an operating CCIS that conforms to the architecture. It does, however, provide a structure or framework on which a designer can combine the functional representation and standards to form a design basis for a specific future CCIS.

The architecture is *generic* in the sense that it describes no single CCIS but a family of systems. Through this generality, future systems built on the architecture will be able to support particular applications, systems, and network capabilities that are available and selected at the time a particular system is implemented. Since a CCIS must be realized within acceptable cost and time constraints, at several points within this section the ideal is tempered with practical considerations. Building an operational CCIS will require use of modern technology—up to date but proven in practice. An implementation cannot be made to depend on unproven research. Hence, the architecture could be implemented with tech-

nology in use today or that is a direct extrapolation of known methods. When future methods or devices are proven effective and useful, they should easily integrate into the CCIS. Supporting modern technology is important, but the ability to have an incremental growth path and to support interoperation with existing systems is equally so. Evolutionary growth and the planned upgrade of existing systems are overall requirements for military command and control systems.

The architecture is intended to be *scalable* in the sense that all nodes will not be identical; neither will all network links be identical. Not every feature or capability will appear uniformly across a CCIS. No specific behavioral specifications are given in the architectural descriptions. It is incumbent on the architecture to be able to adapt to quantitative specifications when they are made, perhaps by evaluating the requirements of an application or perhaps by observing the capabilities of an existing processor or communications link that is to be part of the CCIS.

The generic architecture together with the target profile of Section 4 provide the information that would be needed by a designer working on a specific CCIS implementation.

3.2 TECHNICAL IMPLICATIONS OF THE ARCHITECTURE

The term CCIS has two components, "command and control" and "information system," each with its needs, goals, and requirements. The command and control requirements were covered in Section 2. Here, the information system technical characteristics necessary to meet the desired CCIS characteristics are examined. They form the basis for the decisions made in defining the generic architecture so that any CCIS built to conform to the architecture will possess the desired technical characteristics, and will, therefore, provide the desired CCIS characteristics. The order of the sections that follow mirrors the presentation of Section 2, but the emphasis here is on the technical issues rather than the CCIS needs. Security, however, is covered in Section 3.10.

3.2.1 Interoperability

Interoperability is a characteristic shared among information systems, not a single value, but a set of states from non-interoperable systems that cannot communicate with each other in any way to tightly coupled systems that can share resources. Interoperability is considered at three levels, communications, data, and process, from least to most capable.

Communications interoperability is the ability of two systems to pass messages and share other forms of communication. The simplest instance is at the level of character interchange, for example where all components use the ASCII character set. The most common example of communications interoperation today is electronic mail (e-mail) that can bridge a variety of public and private networks. Future systems will support a multimedia e-mail that includes voice and video. A natural extension includes simple file transfers, given appropriate access privileges. For systems that are organizationally distant and have no need for closer cooperation, communications interoperability may be all that is required. Communications interoperability requires some form of common data structure standards to be used by the application. For example, when common document formatting standards are used, a document developed on one system can be exchanged with another system so that a user on the second system can process the document as needed. Similarly, communications interoperability supports the use of electronic data interchange (EDI) techniques (provided the systems involved subscribe to the same standards). Thus the underlying technical requirement for communications interoperability is the use of common data communications protocols and applications structured to make use of them.

Data interoperability gives the ability to exchange data so that the data to be transferred can be correctly interpreted by the recipient. Alternatively, the systems may share data element definitions and have common data structure standards. Also included are databases shared in such a way that an update at one location will automatically change any backup copies of those values at another location. The use of common data element standards preserves the meaning and relationships of data across systems. This level of interoperability will be required of CCIS components in units related organizationally but geographically dispersed. The underlying technical requirement for data interoperability is the mechanism for transmitting or sharing structural information.

Process interoperability provides the ability to share system or application processes or services. For example, an application running on one node might start a process on another node. It might directly access a remote database and return the results of a query to a third node. Such systems tend to be tightly coupled and make sense from an operational perspective where communications links have high bandwidths, are stable, and support the needed security. Typically, this requires co-location (supported by a common local area network) or permanent facilities (with access to high-speed wide area networks). The technical requirement for process interoperability is a distributed or network operating system that supports, for example, a client-server multi-tasking structure.

3.2.2 Survivability

As described in Section 2, mechanisms that ensure survivability include flexibility, mobility, and redundancy. Flexibility and mobility are covered below. Redundancy applies to all parts of the CCIS: communications, computing equipment, and data. Redundant communications circuits require intelligent network management that can dynamically select the best routing for the traffic at hand. Redundant computing resources suggests distributed computing based on multiple computers in preference to the large, central processor approach of today's CCIS. Redundant data requires automatic mechanisms for replicating and distributing data to alternate and backup locations where a given function might need to be performed. In addition, the ability to capture backup information in real time is required.

3.2.3 Flexibility

A CCIS needs to be flexible with respect to many factors. Environmental flexibility is largely a hardware issue, not addressed in depth by this effort. If gaining hardware flexibility leads to a variety of computing platforms, transferring applications may become a problem. Use of standard programming languages such as Ada will enhance this flexibility.

Flexibility with respect to force organization, structure, and operational level requires the CCIS to be reconfigurable across a range of functional alternatives. Such flexibility requires standardized, general-purpose computing platforms so that applications can operate on any platform. In addition, users need to be familiar with all the applications their assignments might require. This can be greatly aided by standard user interfaces that make behaviors that appear similar—even if parts of separate applications—work the same way.

Flexibility with respect to threat is closely related to survivability. It calls for the ability to perform any function from any of several different locations. As the political and military structure of the world changes, policy and doctrine must keep pace. The technical requirements are largely the same as those for interoperability, with the added requirement that interoperational relationships must be dynamic. The ability to establish new links, particularly for data interoperability, is essential. Other changes in doctrine may alter the way data is organized and disseminated.

Flexibility with respect to technology requires the ability to accept incremental software changes, evolutionary introduction of new technology including hardware compo-

nents, minimal disruption of operations due to new training requirements, and the ability to operate within a variety of configurations. To make this possible requires an accepted set of standards that are well supported by vendors of both hardware and software.

3.2.4 Mobility

CCIS mobility requires a highly flexible relationship between computing and communications resources. Operational functions originally connected by fiber optic networks may at a later time need to be connected by low bandwidth radio links. Mobility requires interoperation strategies that do not depend on continuously available computing and communications. A message or database update sent from one location to another may need to be delayed for various reasons. Mobility as a countermeasure may sometimes require electronic silence, during which the system would not be able to send its regular data traffic or acknowledge receipt of traffic received. One impact of this is on the utility of communications protocols and data management techniques that require positive confirmation before allowing other events to proceed.

3.2.5 Affordability

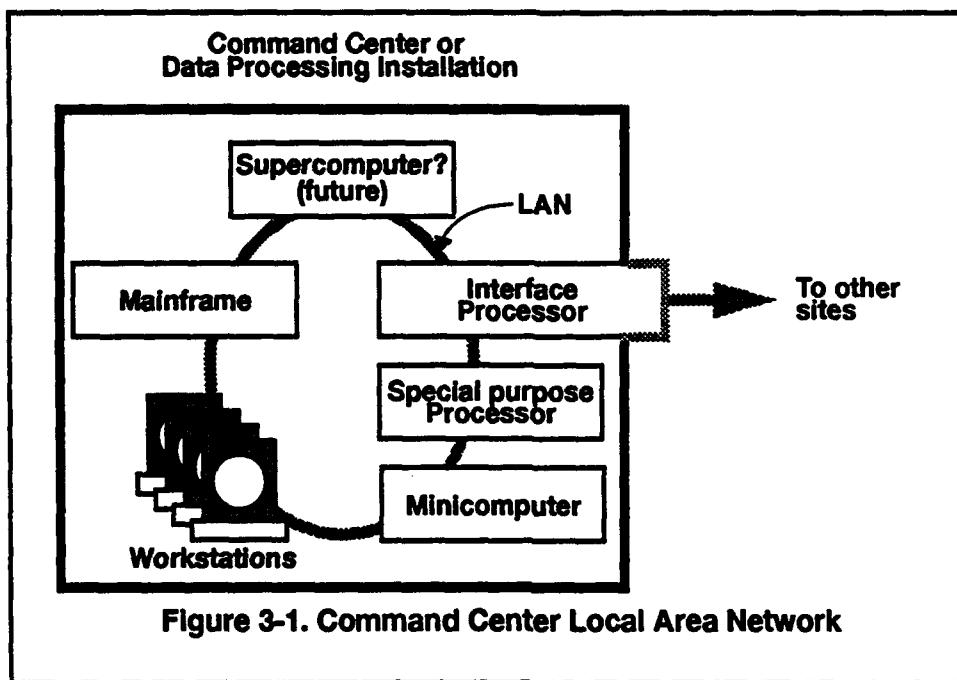
The best way to lower the cost of hardware and software is to buy it off the shelf rather than build it. Most commercially available software is written for standard platforms consisting of a computer architecture, operating system, communications, database management, and user interface. Today, such platforms are largely proprietary, but there is considerable effort being spent on developing internationally accepted, nonproprietary standards that specify the interfaces between the parts. Building systems that implement these standards is the technical route to cost avoidance. As standards continue to make the software independent of the hardware, the greatest opportunities for reducing hardware costs by increased competition will arise. Even when special militarized, hardened equipment is required, hardware-independent software will provide for the maximum degree of competition in hardware procurement.

3.3 OPERATIONAL POLICIES

Within an information system, certain abilities and behaviors are defined because they are the only way to meet mission requirements while some reflect decisions made by designers or architects concerning the structure of a system. In either case, the decisions reflect policies that determine how systems are organized and how they operate. The following operational attributes apply to any CCIS that satisfies the architecture.

- a. **Distributed operation.** The CCIS consists of independent processors that can work autonomously but are potentially most effective when cooperating.
- b. **Distributed data.** The data required for any application may reside in distinct "chunks," in geographically separated locations.
- c. **Concurrent operation.** Groups of applications can share computing resources. Single applications may simultaneously require several possibly widely separated computing resources. To achieve either of these requirements, synchronization among cooperating processors is required.
- d. **Dynamic configuration.** Normal operation of the CCIS will permit the network to survive and be effective even though it may be broken at any point and certain nodes may disappear, possibly to request a reconnection at any place or time.
- e. **Heterogeneous environment.** Interaction between hardware and software systems potentially of greatly different structure is required.

CCIS users will be spread across the world in fixed and mobile sites. Each may be supported locally by its own distributed computing system as illustrated in Figure 3-1. Data needed to compose a message, to run a simulation, or to develop an operations plan may have to be gathered and integrated from sensors and databases at several sites, perhaps half a world apart. The architecture reflects this distribution. Its most abstract representation is a series of interconnected nodes that communicate, passing commands, messages, and data. A more detailed picture is given below in two levels of detail: the top-level functional architecture (connectivity and computational functionality) followed by a breakdown of each of the major areas. These areas provide the functionality needed in the design and implementation of any specific CCIS. Modern information systems are based on high-level abstractions that have already proven their value at the architectural level. Four abstractions that will be referred to in the following sections are as follows:



- a. Client-server model. A part of a system that requests a service from another is considered a *client*, while the provider is a *server*. For example, a client application program will request information from a database server; a client node will request a network server to transmit a message. The model assures that the requestor does not need to know details of how the service is provided.
- b. Autonomous agents are programs or processes that perform actions on behalf of the CCIS without being initiated by or associated with any particular application or user process. For example, a “data archive agent” can monitor file access and automatically “mirror” file updates to a backup store, aiding in fail-soft recovery. Agents can provide a mechanism for implementing an active system security monitor.
- c. Object orientation. Object orientation is a design paradigm that considers a system in terms of the objects that are manipulated (messages, plans, maps, etc.), the attributes of those objects either individually or through membership in some larger grouping (security classification level, for example), and the sets of operations defined on those objects.

- d. Peer-to-peer communication. As indicated in Figure 3-2, communication between segments of a system should behave as if it were proceeding from any given level in the sender to the same level in the recipient.

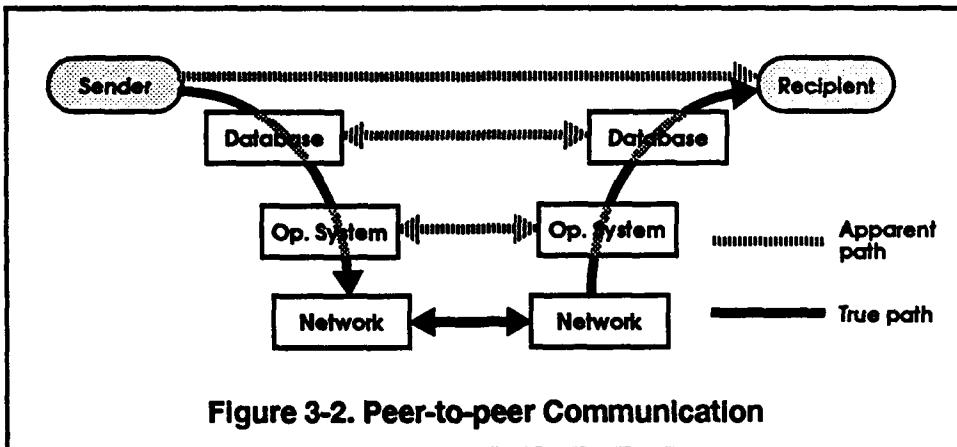
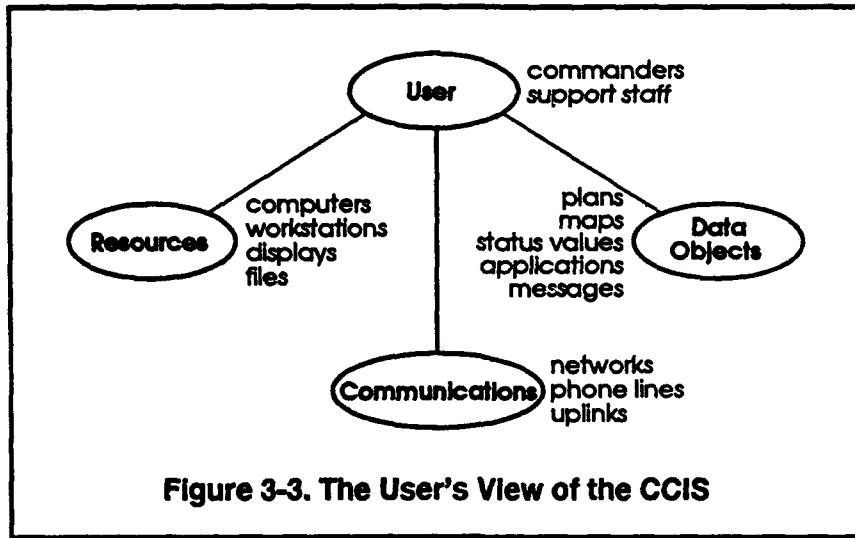


Figure 3-2. Peer-to-peer Communication

Although the characterization of these abstractions is recent, the underlying principles are not new. The client-server model is the oldest of these. Many systems have been built on this principle. Agents, under one name or another, have existed within operating systems since the 1970s. During the 1980s they evolved into the network domain. Object orientation is based upon principles of data abstraction that date back to the 1970s [Morris 1973]. Peer-to-peer communication is a natural extension of the layered network open systems model described in detail below. The abstractions are primarily used as a descriptive convenience. Although all have been proven in actual implementations, a CCIS system designer may choose to structure a specific CCIS by implementing the abstractions directly or by choosing an alternative method that yields equivalent behavior.

3.4 THE USER'S VIEW OF THE CCIS

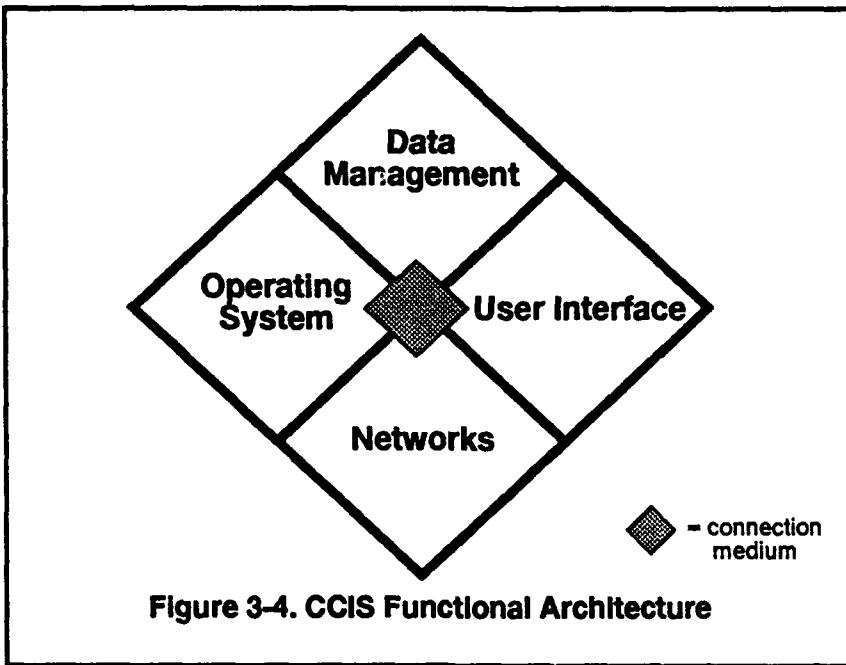
The user's view of a command and control information system appears in Figure 3-3. A user pursuing a C² mission will use objects, communications, and resources needed to achieve a task at hand. Communications include those local to the user's node as well as wide area networks and gateways to other networks. Supporting the user are resources provided by local and remote host computers and their operating systems. Data objects within the CCIS are, for example, images, operation plans, maps, and messages used within a single node or transmitted to other nodes around the network. Applications such as JOPES are



included among the data objects. Just as data elements can be retrieved and used, so can applications. Users are, of course, the "personnel" of the JCS PUB 1-02 definition quoted earlier. Included in the class of users are not only those responsible for the C² mission but also those who administer the system, develop the applications, and maintain the databases. Understanding that administrators, developers, and maintainers are among the class of users is important: unless systems are developed and maintained in a setting virtually identical to the operational environment, subtle errors of behavior, performance, and correctness can go unnoticed until they manifest themselves, perhaps during a crisis.

3.5 THE CCIS FUNCTIONAL ARCHITECTURE

The top-level architecture of the CCIS, as illustrated in Figure 3-4, has a structural resemblance to the user's view with the following differences: the user is replaced by a user interface, the local resources by the local, distributed, and remote operating systems providing the node's support environment, the data objects by the data management, and the communications by the local networks and gateways to other parts of the CCIS network or to other networks. The architecture is functional in that it reflects a breakdown by functional area rather than a mapping of a physical structure. Basing the architecture on function in this way assumes that functions are always clearly separable, a simplification of the true picture, but one useful for exposition. In practice, boundaries are not as cleanly separated as the diagram indicates. The boundary between a network and an operating system, for example, is often fuzzy. The ISO seven-layer network reference model (ISO 7498) explic-



itly includes functions conventionally performed by an operating system. Any implementation will likely require some form of network distributed operating system in which core operating systems functions and networking functions are intermingled. Based on this simplified diagram, however, the internal connectivity of the architecture can be seen: a user, working through the user interface can use local resources and local data. Through the network, remote resources and data can be obtained. The operating system can use data management to store and retrieve information relevant to its operations and the network in support of operations shared with other processors or nodes. Similarly, data management can use local operating system services and it can communicate with data management elsewhere on the network.

The operation of the CCIS is based on the application. Although the generic architecture is independent of any *specific* application, it takes into account the central role of applications. A typical application might be an entire planning subsystem such as JOPES or it might be a simple user-to-user electronic message transfer. The C² user will have a library of support programs or subsystems. But other users, those who maintain and support the CCIS and its databases and applications, will use the same nodes and networks. The applications available to the developer and maintainer are software engineering tools: compilers, text or program editors, and configuration managers. System administrators have database and network maintenance functions as their particular application programs.

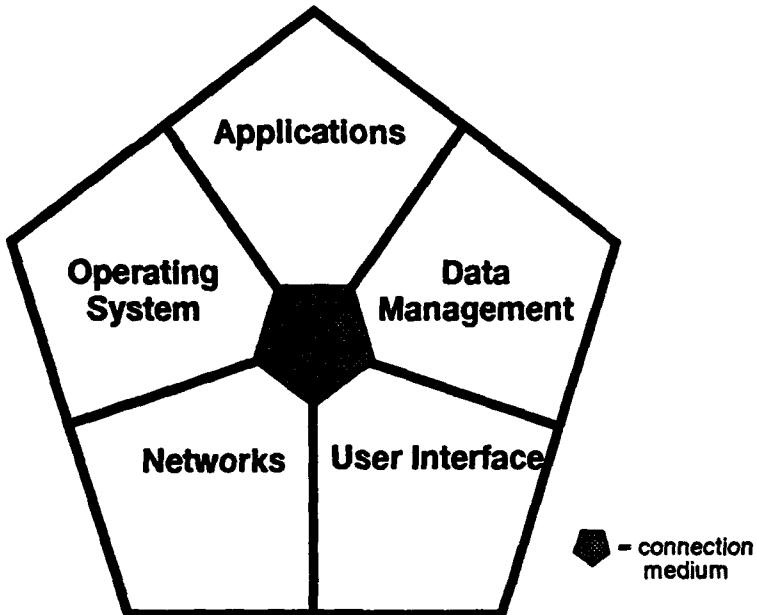


Figure 3-5. CCIS Functional Architecture with Applications

Figure 3-5 adds applications to the basic architecture. The interconnecting lines signify that, while an application communicates with the four functional parts of the node structure, these may communicate with each other in supporting the application, the user, and the CCIS itself. For example, data management will interact directly with the operating system and network to perform periodic backup of data, independent of any particular application. Communication from one functional area to another is mediated by services that provide interchange mechanisms and standard interfaces at the connection points. The diagram of Figure 3-5 is an ideal. In practice, applications will embed other applications (as, for example, a planning application may contain a graphics subsystem); they will communicate directly together without recourse to external services such as those provided by data management, operating system, or user interface; and they will even contain within themselves functions that duplicate in part these external services. Duplication is a necessary consequence of the thrust towards use of COTS products and the need for a new CCIS to provide an incremental growth path from older systems.

3.5.1 Standard Services and Interfaces

Broad use of standards is basic to satisfying requirements of interoperability, software and data portability, consistency of the user interface, maintainability, extensibility, and flexibility. Standard methods, interfaces, data definitions, and protocols provide a framework within which a designer can make the implementation decisions for a CCIS. Based on the generic architecture, the designer of a CCIS will take into account new, updated, or transferred functions, transition plans, interfacing, security, physical environment, and the anticipated skills of users, to define a set of detailed system requirements. Using the available technology, code transferrable from older systems, and commercially available programs, components, and subsystems, a system design will evolve. Without standards, each CCIS, subsystem, and application will require new, probably unique, certainly incompatible, interfaces and protocols. With established and tested standards it will be possible to avoid the burden of complete interface redesign each time a new piece of equipment is added or a new application installed. The standards aspect of the architecture is characterized by seven areas based on the characterization developed by NIST in the Applications Portability Profile [NIST APP/OSE 1990]. They serve the following functions.

Data management supports the storage, control, distribution, management, and allocation of simple data such as text and numeric information and complex items such as complete documents, maps, charts, images, and multimedia objects. Information about the network and nodes themselves (e.g., configuration and routing data) may be also handled through data management services.

Network services are the transmission and interface standards and protocols that support logical and physical communication. They describe and constrain how the hardware and software of the nodes cooperate in node-to-node interaction.

User interface services support visual and functional interaction with the user, providing access to hardware and software and graphical user interface (GUI). They control the presentation of data and mode of interaction.

Operating systems manage hardware and software resources and software interfaces, including local and distributed execution of application programs. Standards in this domain include those that cover program-to-program communication and synchronization as well as management of memory and interfaces to network and data management services.

Security services provide for the privacy, protection, and integrity of the programs and data that make up the CCIS. Security pervades the model, applying at every interface and point of data transfer. Security has four aspects:

- a. *Authenticity*, assuring that a given service requester is indeed the person (or process) whose identification and passwords are being presented.
- b. *Availability*, assuring that data and program resources are delivered promptly to a legitimate requester.
- c. *Confidentiality*, assuring that data and program resources can be obtained only by users and processes that have valid clearance, authorization, and need-to-know. "Access control" is included here.
- d. *Integrity*, assuring that data and program resources are stored, transmitted, and used without any corruption or accidental or unauthorized change.

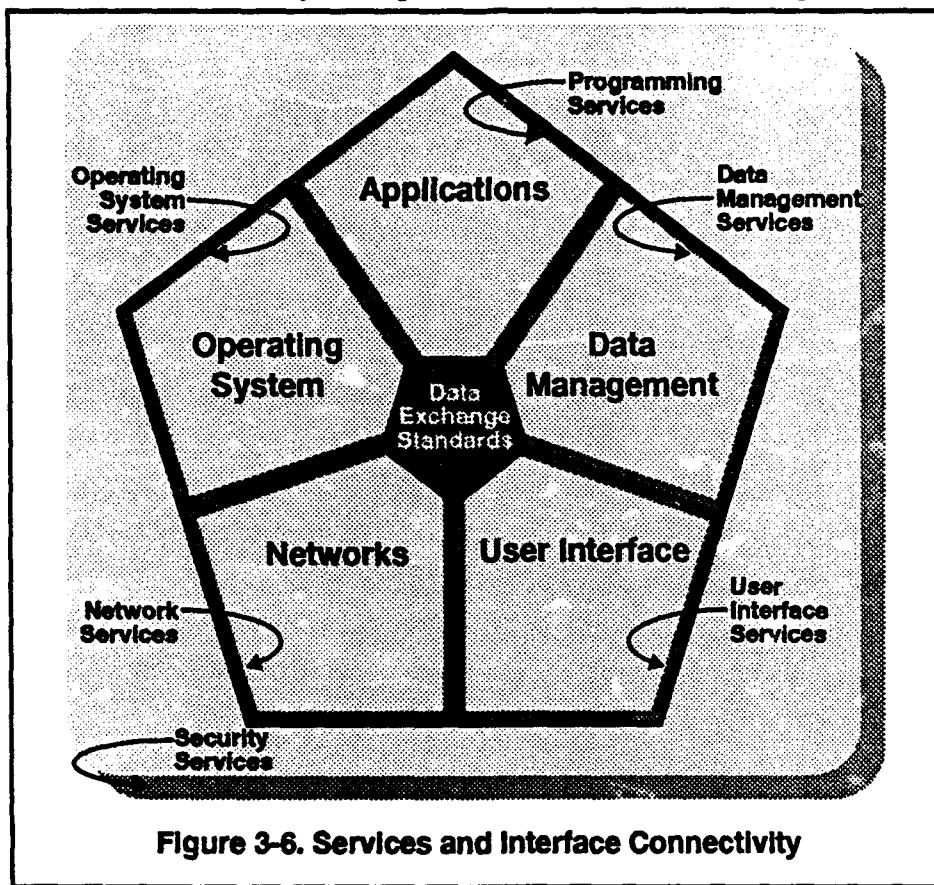
Programming services affect two aspects of the CCIS: application development and the execution of applications. In the former case, programming services support the development, checkout, installation, maintenance, and testing of application and system software. At present there are few standards to support this domain although work on software engineering environments, specification languages, and program development and maintenance tools is moving toward the stage when standardization is likely. In the latter case, the architectural role of programming services is in the language-specific and interlanguage interfaces that support the client-server interaction.

Data exchange permits the interchange of data among applications and systems, and display of data to the user in a way that preserves the meaning and relationships, for example, the electronic transfer of documents between systems or applications, preserving their form and appearance as well as their content. (Excluded are standards related to the physical data exchange of data such as disks and tapes.)

Linked with all of the above are *management and administrative services* that support the monitoring, maintenance, configuration management, and self-diagnostic activities that go on continually while the CCIS is in operation. Database and network administration are also included here. Those few standards that address administrative services are covered in the service area descriptions below.

Within the functional architecture, the relative position of data management, network services, user interface, and operating system services—and the interfaces they support—is clear and well-defined. As indicated in Figure 3-6, they support the interchange of infor-

mation between an area and any other part of the CCIS. The remaining areas are different



in nature. Programming services support the development and maintenance of applications. Once an application is complete, tested, and integrated into the CCIS, only a subset of the programming services is required. The run-time aspect of programming services is what appears in the figure surrounding the application. Security is pervasive, distributed throughout the CCIS protecting the integrity of the system statically, dynamically protecting the transfer of objects from point to point within a node and among nodes on the network. It cannot be well represented in a simple diagram but is shown in Figure 3-6 as a grey background or substrate in which the entire architecture is embedded. Data exchange standards are static, applied during the implementation of the applications and system programs at the time data structures are defined. They affect the transfer of data from application to application and the presentation of data to the user, but are actively involved in the CCIS operation only as they are compiled into the executing software and applications programs.

3.5.2 Practical Aspects of the Architecture

Some aspects of CCIS structure and operation do not easily fit the architecture and services model. Although they do not appear at the top level of the architecture, fault detection and recovery, for example, are omnipresent at the operational level. Although directed more towards implementation than architecture as such, it is important to keep them in mind. The following items summarize some of these practical considerations.

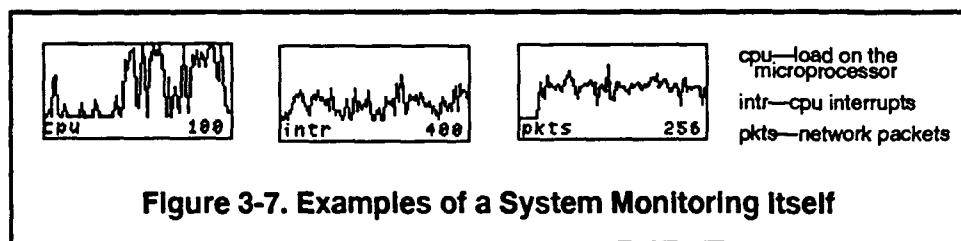
Interruption of service. It is not possible for an architecture to cater for all possible threats and faults that might occur, although, of course, an implementation must consider those that are most significant. Shielding against EMP (electromagnetic pulse), nuclear hardening, and protection from enemy action against cables and satellite links are required in wartime. The architecture anticipates the possibility that links may be broken and allows for redundant linkages such as, for example, redundant ground-wave communication, but a total threat analysis would depend on much more detailed knowledge of design considerations such as configuration and operational environment.

Reconfiguration. For purposes of security and to recover from service interruptions, the networks of a future CCIS are planned to be able to reconfigure themselves. Although reconfiguration is an attribute of implementation strategy rather than of architecture, it must be anticipated in the architecture.

Integration of COTS. It is a principle of design for future CCIS implementations (as well as for most new DoD software-based systems) that commercial packages will be used wherever possible. The benefit is clear: software will have external support, it will be developed and documented by the suppliers and not at government expense, and there will very likely be a base of training and trained users. However, unless these COTS products use the standards and protocols specified for the CCIS, integrating them into the system will require special programming. Most commercial software products carry their own environment and user interface with them. Integrating them smoothly into a CCIS that requires common behavior and interface may not be possible in the near future. The problem is compounded in the programming area, where proprietary environments are common.

System monitoring and maintenance. In operation, the CCIS will have to monitor its own operation both in terms of the existence of connectivity and in terms of performance. By keeping track of activity on the network and within the processors, the CCIS maintenance function will be able to determine both whether the system is operating correctly and also determine if the capacity of the system is adequate for the load. As an example, Figure

3-7 illustrates three examples of self-monitoring from the networked workstation upon which this architecture description was written. The first diagram tracks use of the central



processor, the second is a record of processor interrupts, and the third shows the number of packets sent, received, or passed through on the network. Such information can drive the self-monitoring process. System security can be enhanced through self-monitoring as well. Given expected profiles of load and activity, the CCIS can look for abnormal patterns of behavior on the networks or in the processors and pass exception information along to a human or computer security agent. The networks can be designed to shut down or to isolate critical segments or resources such as databases under conditions of unexpected or abnormal activity.

3.5.3 Parallelism and Synchronicity

Network operations are highly parallel; nodes do not operate in isolation. The earlier definition of a node as “where a user does work, usually gaining access to system resources through a personal computer, workstation, or some specialized piece of equipment,” is true but limited. Within a given node, many users may be operating simultaneously, cooperating on a problem or working independently. A given application may have several output devices—high-resolution graphics, character monitor, and audio, for example. Each of these would have its own set of control programs, providing a single user with three instances of the user interface.

The single user view of the CCIS node is a useful abstraction. In some nodes, a group of users will share resources. A near real-time multiprocessing capability will be required at the node level. While novel system and processor architectures make near real-time processing of distributed applications possible, advances in display and processor miniaturization are making it likely that single hand-held systems may eventually participate in a CCIS as full-fledged nodes. A recent item in the press showed a “Soldier’s Computer” consisting of a one-pound pocket computer with voice/data radio and global positioning receiver.

Attached was an instrumented helmet with "heads-up" display, earphone, antenna, and an attached hand-held joystick [Washington 1991, 18]. The capacity of a network to support a large complement of such nodes will likely require a significant increase in bandwidth.

Applications themselves may take on broader scopes. In addition to many "single-thread" activities occurring in parallel, some C² functions will be served by processes that use servers in different locations on the network. Although an active database may reside in a single location, full or partial copies may reside at different places on the network. One data server should maintain the master copy. To avoid overloading a host processor or the network links that serve it, geographically distributed users of the database may work from subsidiary working copies. The master copy will then be used as the basis from which simultaneous parallel updates will be distributed. The number of very powerful, high-performance computers on any CCIS network will be limited. Thus, an application that requires both database service and powerful computation may execute through coordination of general and special purpose processors at different points on the network. Cooperative activity of this sort is typical of the distribution of functionality across a network.

3.5.4 Cost Considerations of the Architecture

Cost is normally associated with a system design or implementation rather than with an architecture. When a specific system is designed, it will require a cost study taking into account technology and performance requirements at that time. The following cost-related aspects of the architecture should be mentioned.

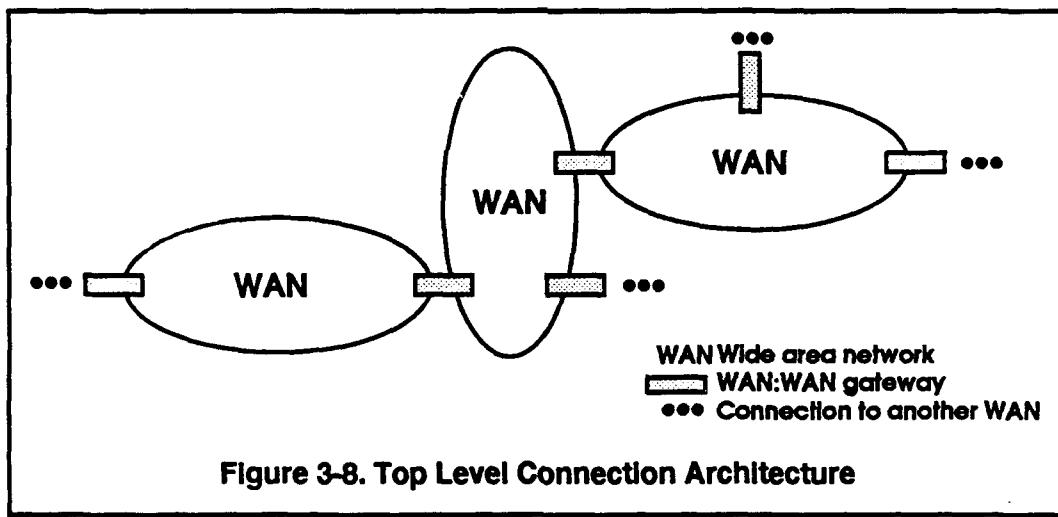
- a. It is aimed at state of practice technology. Potentially expensive and risky research results will not be required to implement the system.
- b. It suggests use of open systems standards. If this guideline is followed, there is reduced likelihood that future enhancements will require specialized hardware and software to be developed.
- c. It recommends standards that are being adopted widely throughout the commercial as well as the federal and military domains, hence the likelihood that COTS products can be smoothly integrated into a CCIS built to the architecture.
- d. It builds on a structure already proven in commercial systems, hence the likelihood that a design based on the architecture can be implemented using known methods. The high-level architecture of Figure 3-5 could easily describe the

functional structure of networked CAD (computer-aided design) systems heavily used in the open market for years.

3.6 NETWORK SERVICES

Although connectivity is but one aspect of the CCIS architecture, it is fundamental to CCIS operation and it is central to the top level and the outside view of the CCIS. In the following discussion, terms such as "local area," "wide area," and "gateway" are used. These terms are not intended to constrain technology to today's practice. Rather, they serve as convenient placeholders. Local area refers to connectivity within a single node or site—perhaps within a kilometer diameter—and wide area to any longer range connection. A gateway is any mechanism that interconnects or bridges networks.

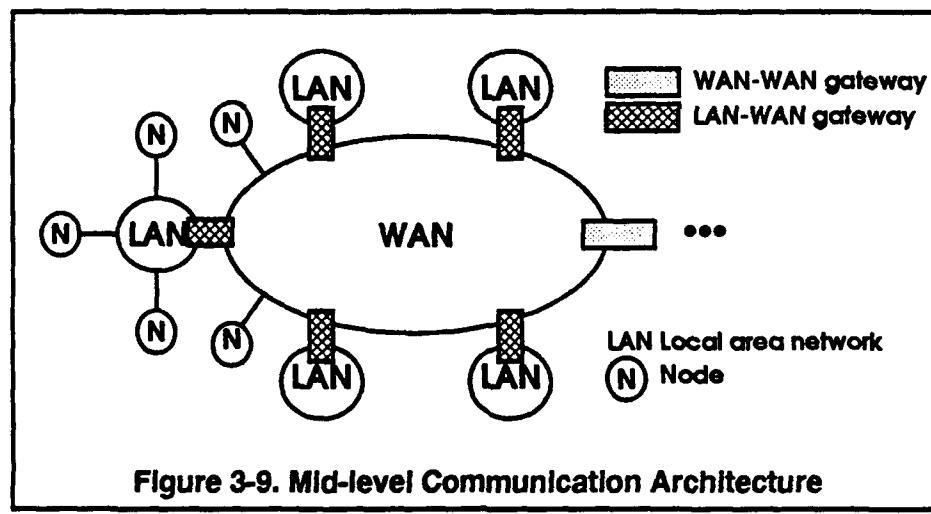
Structurally, CCIS connectivity is constructed of nested networks; the highest level of which is the wide area network (WAN) typified by the oval loops in diagram in Figure 3-8.



In a large-scale world-wide CCIS, there will be more than one wide area network; gateways or bridges will interconnect these networks. Gateways are bridges, places where information is transferred between networks. The gateways shown in the figure are not necessarily single point connections. They might be designed as a group of parallel interconnections using differing implementation mechanisms (e.g., microwave, hard wire, and ground wave) providing for redundancy, fault tolerance, and increased breadth in transfer path. Any of these might be implemented, for example, as a physical connection between two carriers in a given place or as a satellite link across the world. A gateway may be static, always con-

necting two specific wide area nets, or it might be dynamic in the sense of being able to make connection upon demand, dynamically selecting the most appropriate routing. Gateways may be configured between any pair of networks or among any group, allowing any sort of star, ring, or graph network topology. It is possible that the network of the future will not have distinct, discernible structure in the form we know it today. Speculative projections refer to a "cyberspace," a sort of electronic ether into which systems may tap from any point. Although explorations of cyberspace are at present in the realm of speculation, it is technologically feasible to build such a system today—though the cost with present technology would place it far out of reach. The societal impact of such universal connectivity can barely be guessed at, the security problems overwhelming.

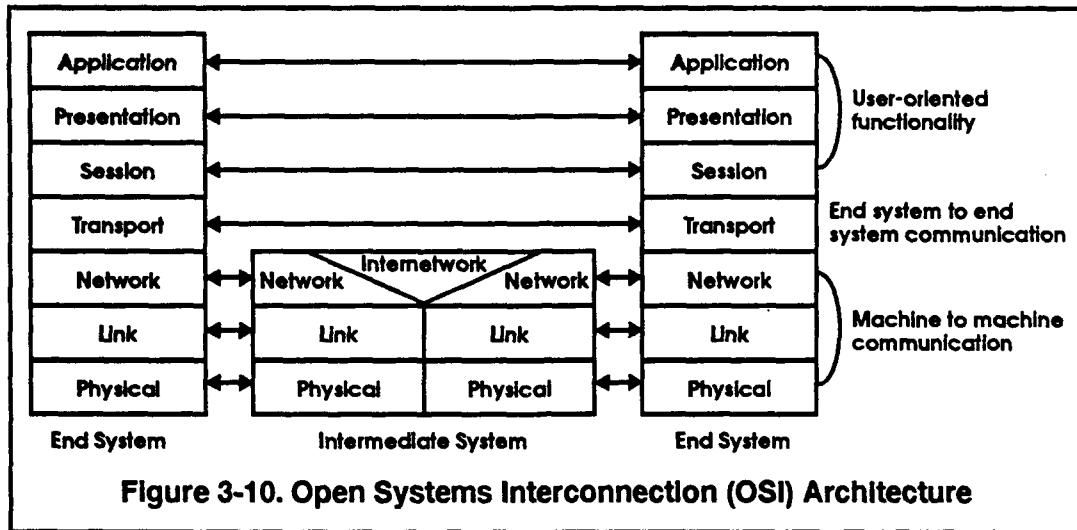
The local area network (LAN) is the basic mechanism for interconnecting individual nodes. While some nodes may be connected directly to the wide area network, in general, nodes physically near each other (distances measured in tens or hundreds of meters) will be grouped into a local area network. Each wide area network will connect a group of these local area networks as illustrated in Figure 3-9. In effect, the total CCIS framework of inter-



connected networks forms a *network of networks*. Telecommunications technology has made such configurations practical. Implementing multilevel security in so complex a structure, however, is considered by many to be beyond today's state of technological practice. Nevertheless, it is an absolute requirement for future defense systems:

Secure networking at multiple security levels is critical to the success of the Integrated Tactical-Strategic Defense Network (ITDN), Defense Message System (DMS), and the Integrated Communications Architecture (ICA) [ASD/C3I 1990].

Although the future of network technology is unpredictable, GOSIP and its descendants are likely to dominate the structure of federal and defense networks for several decades. Therefore GOSIP is taken as the basis for the CCIS network services. Vendor-specific implementations of data communications protocols lead to isolated domains of information that are difficult and expensive to bridge. GOSIP defines standard protocols based on the OSI model, that allow the interoperation of systems of different manufacture. Adopting the GOSIP model for the generic CCIS brings with it another architectural level. Each active



node on the network will have a breakdown of function akin to that shown in the tall stacks on the left and right of Figure 3-10. There may also be so-called intermediate systems acting solely (in this context) as network servers with the abbreviated structure shown the middle of the figure. An implementation will need to define the allocation of function between end-system functions and intermediate-system functions. The OSI architecture helps to illustrate the fuzzy line between the functional areas of the architecture. It is very likely that such communication-oriented applications as e-mail systems will embed application layer, presentation layer, and even session layer functions. The layering of the OSI model will help in defining interfaces with existing systems, a process that will be required during the many-years transition from present-day WWMCCS.

3.6.1 Network Operation and Reconfigurability

A simple computer network will use coaxial cables or optical fiber physical links to connect the nodes with each other. More widely based networks may use telecommunications links to handle long distance segments. It is characteristic of such networks that the connectivity, once established, is fixed. Although the failure of any node or link may not make the entire network inoperable, to make any change to the network usually requires some sort of manual reconfiguration, often requiring changes to network operating system tables or even, in older systems, changes to the system hardware or software itself. Network management for a CCIS must be significantly more adept. As with commercial telephone networks, the CCIS network must constantly monitor traffic activity and be prepared to reconfigure itself as need arises. When considerations of node mobility, security, and the need for nodes to disconnect and rapidly reconnect are added, integration of telecommunication and data transmission facilities will be required to an extent far beyond today's commercial endeavors. Although with the advent of cellular telephony, commercial networks have begun to address mobility, the lack of security in the commercial cellular telephone network makes that technology in its commercial form inapplicable to the military CCIS. It is necessary to assure that if part of the physical network is compromised, the breach can be detected and the remainder of the network be able to reconfigure and operate effectively.

In the future, a CCIS network will be able to operate independent of any particular node or control computer; there should be no central single failure point whose loss or isolation will put the entire network out of operation. Any node that contains a general purpose processor and has connectivity to the network should potentially be able to restart, monitor, or administer network functions. Whether any specific node will be able to do this or not is an administrative policy decision. To achieve this decentralized control, the network should be thought of as operating autonomously (with respect to any node), controlled by autonomous agents that independently handle routing, configuration, monitoring, and maintenance functions. However, since this is precisely the mechanism used to propagate computer viruses, significant self-checking and validation will be required. Although loss of any particular node may terminate the network control processes running on that node, the agents running on other nodes will be able to keep the network running and, when the node returns to activity, any existing agent can, subject to security policy restrictions, restart the network process on that node.

3.7 OPERATING SYSTEM SERVICES

The distribution and connectivity of the CCIS network leads to architectural decisions that must be made with respect to the operating system. An operating system in an automated CCIS provides a means for managing resources and the interfaces. POSIX and its descendants would appear to be the dominant specification for operating systems in C² systems over the next several decades, but it is still under development and has not yet achieved significant success in the commercial market. POSIX is an ANSI and FIPS standard. Not in itself an operating system, POSIX is rather a definition of an interface and environment. Although its primitives are based on those of Unix, it does not depend on any particular Unix implementation. The current version of the POSIX standard defines C language bindings. Work is under way to complete definitions of Ada bindings.

3.8 PROGRAMMING SERVICES

Programming services support the development of software and its execution. The development environment concentrates on services that support the design, implementation, checkout, installation, maintenance, and testing of application and system software. In application execution, the architectural role of programming services is in the language-specific and interlanguage interfaces that support the client-server interaction. From the point of view of the functional architecture, both aspects of programming services fit in at the same place. The diagram in Figure 3-11 shows the run-time part of programming services fitting around the application. At the time of program development—especially that part of the development process during which a new or changed module is integrated into the system—programming services subsume the application in the architecture, in effect *becoming* the application. This is illustrated in part (A) on the left of Figure 3-11. It can be argued that an ideal view of programming services has CCIS applications and systems developed within the CCIS itself, the programmer acting as a CCIS user, the program development environment (PDE) becoming the application. In this way, there is no distinct integration phase of development. Developed within the operational CCIS environment, the completed application is already integrated. Such an ideal will only be met in the future. A great deal of effort is being spent in the commercial market to develop the software engineering environment (SEE), and a large variety of computer-aided software engineering (CASE) tools are available today. One common aspect of these tools, especially the

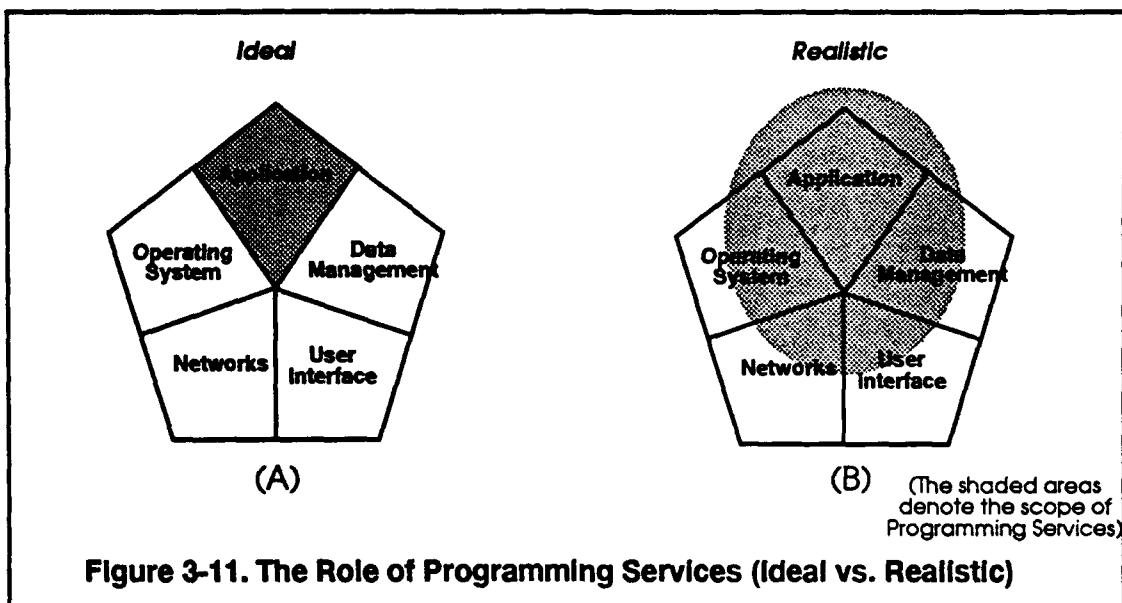


Figure 3-11. The Role of Programming Services (Ideal vs. Realistic)

integrated packages, is that they embed their own user interface and data management. Many even come with rudimentary operating system and networking applications. Because of these embedded systems functions, the program development, maintenance, and integration view of the architecture will probably look more like part B of the figure, where programming services subsume much more of the services area than the ideal would indicate.

3.9 DATA MANAGEMENT SERVICES

The CCIS database, the collection of data used within all command and control information systems worldwide, is, in technical terms, partitioned and partially replicated. To say that it is partitioned means that not all the data of a given kind is stored in one place. For example, unit status data will be stored wherever there are units, and target tracks stored everywhere targets are tracked. It is partially replicated because some of the data is replicated, but the entire database is not stored everywhere. The same examples apply: status information from a unit is sent to adjacent units to keep them informed, and targets tracked by observers are shared with the weapons systems users assigned to attack the targets. The management of this database in the CCIS architecture is carried out by what is known as a "federated database system." The key characteristic of a federation of data management systems is that each maintains a degree of autonomy appropriate to its relationship to the other data management systems it is connected with. At the highest levels, typical of the

relationship between command centers, the degree of autonomy is very high and the federation is described as "loosely coupled." Within a single command center, particularly within a single cell of a command center, the degree of autonomy may be very low, resulting in a "tightly coupled system." In a loosely coupled arrangement, the participants will either have predetermined access rights to data held at other locations or they will have to negotiate for these rights.

Key to meeting the survivability requirements of the CCIS is the additional need to negotiate for replication sites for data. Sites can request to replicate data of other sites (for example, if access is frequent enough), and a site may ask other sites to become replication sites for it. Much of this negotiation is directed by military doctrine regarding alternative locations for functions. If a command center is the designated alternate for another, it would be normal for the alternate to be a replication site for the primary's database. The degree of coupling also affects the type of replication used. Where loose coupling is appropriate, the most appropriate type of replication would be master-slave replication: typically the site where the data originates holds the master copy and other copies are considered slaves. Updates are always applied to the master first, followed by updating of slave copies as independent operations. Generally, only the user at the master copy site will make updates, all other users having only read access. In this sense, the holder of the master copy is the "owner" of the data. There may be a need for distributed transactions where databases at different locations are updated simultaneously. Either all are updated or none are.

The concept of the federated database provides two additional functions at the logical level: location transparency and support for multiple data models. Location transparency means that when an application needs to get data, it need ask only its local data management system. If the data is not stored locally, the data management system will get it from a site that has it, as shown in Figure 3-12. The data manager is a server for data requests of the local client. Logically, communications pass from one data manager to another, but the physical connection makes use of the network services described above. Support for multiple data models means that regardless of the data an application uses, its data management services are provided by the same data manager. In theory, a single query could require access to databases of different data models within the overall set of CCIS databases. When combined with location transparency, these databases could be all at the same site or some at different sites, as shown in Figure 3-12. In practice, this may mean nothing more than the logical idea that the collection of databases appears as a single one. In the general case, every node in a CCIS network will need access to databases at many of the nodes it is orga-

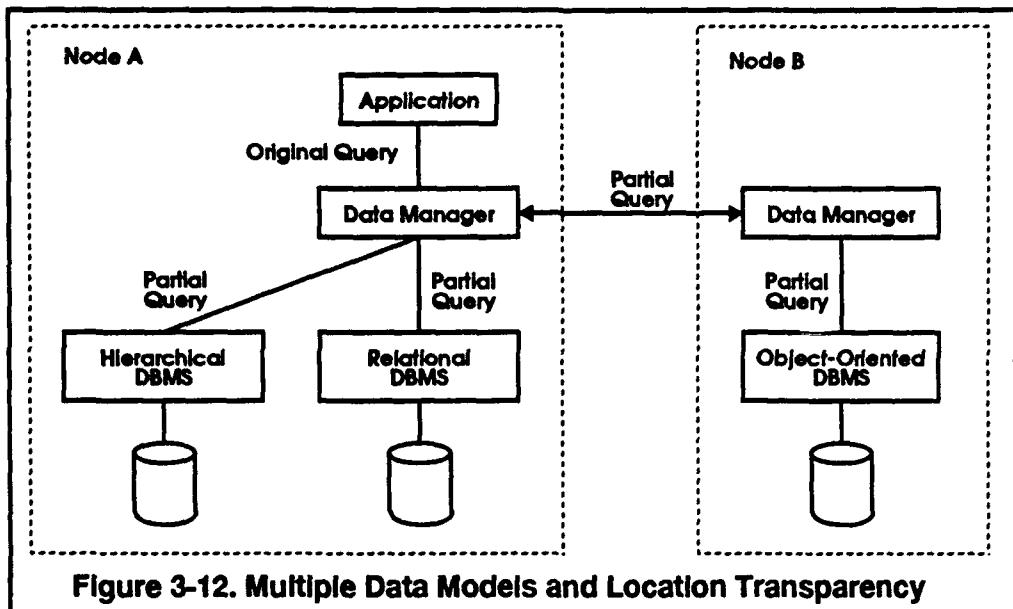


Figure 3-12. Multiple Data Models and Location Transparency

nizationally near. This is accomplished through the federation approach by providing access to the relevant portions of the schema (called an export schema) from the requesting node to the holding node. As a result of the continuity of such sharing, virtually every part of the database is known at more than one location, but no parts are known everywhere. Thus, except in an abstract sense, there is no notion of a single global schema that defines all data and there is no way to access data not known to the data manager at the user's node. Queries and partial queries do not wander around the system looking for a home.

3.10 SECURITY SERVICES

The goal of security in the CCIS is that no information be corrupted or improperly released, that users (human and computer) be properly validated as to clearance level and need to know, and that the validity of the security state of the system be visible at all times to an authorized observer. Standards, indeed, the base technologies themselves are still in development. Table 3-1 reflects current efforts to treat security in the ISO Open Systems Model. A fully integrated assured multilevel security (MLS) in a heterogeneous networked environment may not be achievable within the time period covered by this architecture, but is a goal that should be pursued through incremental advances that introduce increasing multilevel operations over time. Partial solutions are available in the 1995-97 time frame

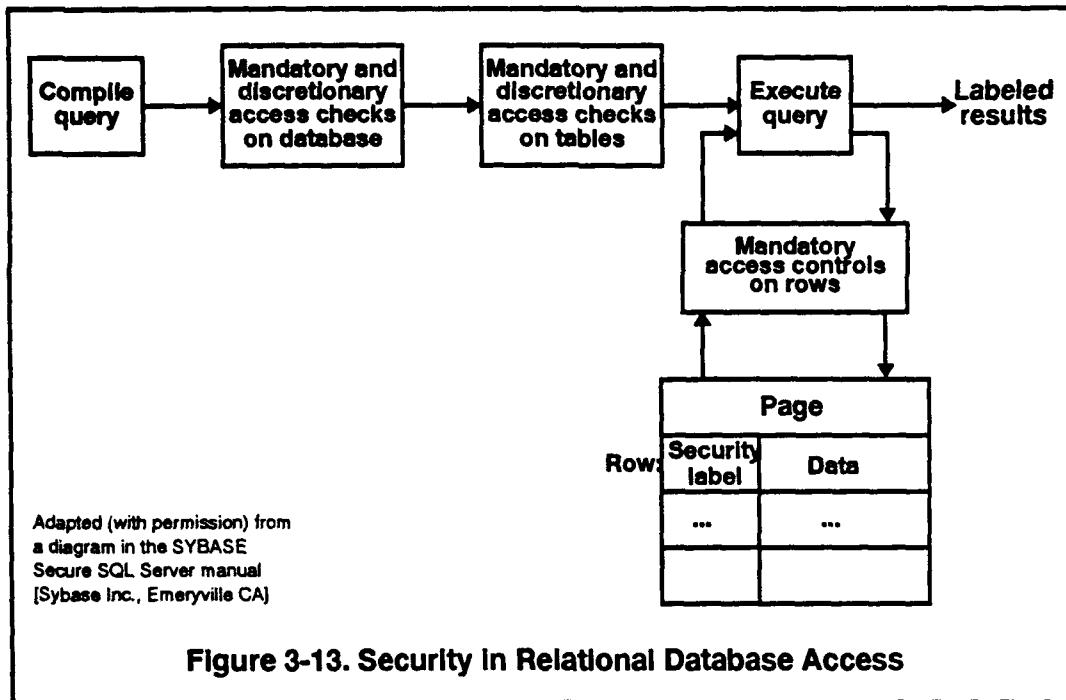
Table 3-1. Security In the OSI Model

Service Layer \n	Authentication	Access Control	Confidentiality	Integrity	Availability
Application	●	●	●	●	●
Presentation			●		
Session					
Transport	●	●	●	●	
Network	●	●	●	●	
Link			●		
Physical			●		

and it is expected that additional solutions will be available over the next 10-15 years as more resources are concentrated on addressing MLS issues.

Before building any CCIS, a security policy to be implemented in that CCIS will have to be explicated. Figure 3-13 illustrates a particular secure access policy for a commercial relational database system. The security is multilevel in that the database, specific tables within the database, and specific rows within the tables may carry their own access checks, in theory permitting users at any clearance level to use the system but gaining access only to permitted data. This structure reflects a particular security policy, one that is close to the state of practice in the early 1990s.

For a CCIS that is to operate in the field for decades, it is risky to "hard wire" security policy into the system. A future CCIS security system should allow the policy to be entered as data and interpreted by the security mechanisms. A policy could be established that would allow the network to divide itself into separate segments at different need-to-know levels with verifiable "firewalls" between the physical compartments. Such freedom brings additional responsibility. A dynamic data-driven security policy, unless unassailably protected, could provide a point for a massive failure of system security. Combining dynamic security policy with autonomous security agents under the control of human security agents might well provide a system that is safe and dynamic. Furthermore, given advances in formal validation of security policy, such a system, although not likely for the 1995-97 target period, might well be implemented within the timespan covered by this architecture.



3.11 USER INTERFACE SERVICES

A computer system's user interface is the means by which the system interacts with the user, presenting and accepting information and commands. More than just a keyboard, mouse, and visual display, it is the totality of all modes of communication with the computer. While graphics standards are not exclusively relevant to the user interface, they have been included in this service area because of the high reliance of the interface design on technology for graphic display.

A good user interface can improve the usefulness of a poor system; a poor one will seriously interfere with the utility of an otherwise adequate system. Because the generic architecture covers systems that will serve users with differing abilities, backgrounds, experience, knowledge, motivations, personalities, skills, and work styles, the user interface is of particular importance and the characteristics of the user interface should be included in the architecture, design, and systems engineering processes from the earliest stages of system development. The provisions of the user interface services are not aimed at a predetermined set of applications or computer and display hardware, but provide a ref-

erence for implementors and indicates goals for developers of future systems. Standard user interfaces can help to achieve a high degree of application portability. They provide a consistent way for those who develop, administer, and use a system to gain access to application programs, operating systems, and various system utilities, but application developers will still design with portability in mind.

The user interface component of the generic CCIS provides for human performance (speed and accuracy), training and retention (time and cost), workload and safety (fatigue, boredom, and accidents), comfort and acceptance, and is intended to minimize the need for special aptitude and skills or extensive training. The following user interface requirements will impact the CCIS architecture:

- a. Separation of the user interface from details of the application, needed for a portable, evolvable user interface, and consistency of the user interface across applications. An application need only accept input and provide output. The user interface transfers input to the program and present results to the user in an appropriate manner.
- b. Provision of an effective interface for human-computer interaction with appropriate use of direct manipulation, graphics, group interfaces, and, where appropriate, multimedia interaction.
- c. Consideration of human factors, including physical and psychological aspects. Physical human factors, known as "ergonomics," concern aspects like glare and fatigue avoidance. Psychological human factors concern aspects such as preventing boredom and the general class of attributes known as user friendliness.
- d. Exploitation of new technology. User interface technology is a fast growing area and advances in hardware and system software will impact the efficient performance of user tasks and will allow new types of tasks to be performed. Research in visualization ("realistic rendering") and related technologies will give significant improvement in the ability of a CCIS user to see the world and the tactical situation as it really is.

There are three principal links between the user interface and the rest of the CCIS.

- a. Graphics Subroutine Library, responsible for graphical information sent to and received from the screen and associated input devices. It can provide primitive graphics functions such as "draw line" and "draw circle" or more complex primitives such as "display overlay on map."

- b. Window Manager, responsible for allocating screen space and input devices to contending applications. It provides for the creation, manipulation, and deletion of windows on a graphics display device, and functions such as repositioning, resizing, and adjusting the depth order of windows. The window manager will also be responsible for mapping a windowed environment into a more conventional interface for devices that do not support windowing.
- c. Tool Kit and User Interface Management System (UIMS), providing tools and abstractions for building user interfaces at a high level, for example, containing abstractions that allow the creation of windows, menus, and scroll bars. A UIMS will help developers to create or use a tool kit, to combine and sequence interaction techniques, and to manage functions such as window management, feedback loops, error handling, checkpoints and restarting. This additional support is intended to facilitate an iterative development approach for semiautomatic construction of graphic-based user interfaces, allowing alternative designs to be cost effectively prototyped.

3.12 DATA EXCHANGE STANDARDS

Data exchange standards, obviously critical for interoperability, are equally important for software portability and operational and acquisitional flexibility by providing common definitions of data types that are shared across nodes and applications. The standards that address these definitions are grouped into the following categories: formatted documents, unformatted documents, graphics, maps and geographic information, meteorological data, video, and audio.

Formatted document standards include the message text formats that define most of the CCIS data communications today, as well as electronic data interchange (EDI) standards becoming very prominent in the logistics area. Unformatted documents are all the documents commonly produced with ordinary word processing or desktop publishing systems today. Maps and geographic information includes topological data, political boundaries, terrain information, information about structures, and any other data commonly found in a geographic information system. Meteorological data includes both observations and forecasts in text and map formats. Video includes still photographs and motion video. Audio

includes voice mail and audio overlays on other data. The types of data to be exchanged will depend on the particular applications that run on the CCIS.

3.13 DESIGN AND EVOLUTION OF THE CCIS

The generic CCIS architecture describes a distributed information system of special and general purpose computers and workstations connected by a variety of direct wire and telecommunications networks, designed to serve the C² mission. At the highest level of design, this is the complete architecture, generic in the sense that it does not apply just to one specific system but to a potential family of systems that might be built over this decade and the next. The generic architecture is, however, too abstract for implementation planning; more detail is needed. This detail is given in the next section where technology and standards applicable to 1995-97 are spelled out in more detail. As new technologies evolve and are reduced to practice, the generic architecture will continue to evolve. However the architecture presented here should stand with minor modifications for at least two decades.

4. WAM TARGET PROFILE: 1995-97

The generic CCIS description in the previous section deliberately omitted specific design information. In this section, the standards and protocols appropriate to a specific CCIS design are presented. They provide a portfolio of definitions, organized into the service areas, with which an implementor will have to be familiar in order to create a design that is open and standard. Where technology is required but no accepted federal, national, or international standard exists, industry, emerging, or de facto standards are mentioned. The exposition here is deliberately brief; Appendices A through G cover the standards in more detail.

A note on abbreviations: Standards are listed using an abbreviated form such as "ISO xxxx," usually without the normal indication of current status since it is expected that many of the ones mentioned will be approved as international standards by 1995. Some emerging standards are commonly known by the name of the group that developed it, for example, the standard developed by the Joint Photographic Experts Group is called JPEG even though it has an ISO designation. A full list of acronyms follows the appendices to this report, but the following recur frequently: ANSI (American National Standards Institute), CCITT (Consultative Committee for International Telegraph and Telephone), FIPS (Federal Information Processing Standard), ISO (International Organization for Standardization), and NIST (National Institute of Standards and Technology). The NIST APP (Application Portability Profile) is also mentioned frequently.

In each of the following sections, a table at the top lists the standards specified in that part of the profile.

4.1 DATA EXCHANGE STANDARDS

Standards in this category are optional, depending on the requirements of particular applications. There is no requirement that systems include applications that make use of particular data exchange technologies. As shown in Table 4-1, some of the technologies described may not yet be reflected in requirements for information exchange. Standards for physical media such as tapes and disks are not discussed in this section except to note that, for the 1995-97 time frame, the most important of these are likely to be for optical systems: rewritable for variable data and CD-ROM (compact disk-read only memory, a data storage technology) and WORM ("write once, read many times" optical disk) for fixed data. For an

Table 4-1. Data Exchange Standards

Formatted Documents:
ANSI X.12. Electronic Data Interchange (EDI).
ISO 9735. EDI for Administration, Commerce and Transport (EDIFACT).
Unformatted Documents:
ISO 8613. Office Document Architecture/Office Document Interchange Format (ODA/ODIF).
ISO 8879. Standard Generalized Markup Language (SGML).
ISO 10179. Document Style Segmentation and Specification Language (DSSSL).
Standard Page Description Language (SPDL).
ISO 8632. Computer Graphics Metafile (CGM).
ISO 10303. Standard for the Exchange of Product Model Data (STEP).
Maps and Geographic Information Standards:
Spatial Data Transfer Standard (SDTS). U.S. National Committee for Digital Cartographic Standards (multiagency working group headed by U.S. Geological Survey).
Vector Product Standard (VPS). Defense Mapping Agency.
Digital Geographic Information Exchange Standard (DIGEST). Digital Geographic Information Working Group (10-nation).
Multimedia Personal Computer (MPC). Microsoft.
Data Compression Standards:
ISO 10918. Joint Photographic Experts Group (JPEG).
ISO 11172. Motion Picture Experts Group (MPEG).
Joint Bilevel Imaging Group (JBIG).
CCITT H.261. Video Coder for Audiovisual Services at p x 64 kbit/s (p x 64).
ANSI. Digital Processing of Video Signals—Video Coder/Decoder for Audiovisual Services at 56 To 1,536 Kbits/s (P X 56).
ASCII (American Standard Code for Information Interchange).

explanation of the distinction between formatted and unformatted documents, see Appendix A.

4.1.1 Formatted Documents

One requirement to support formatted documents in the 1995-97 time frame will derive from the need to interoperate with older systems whose information exchange is based on formatted messages. While this requirement is likely to persist well beyond 2000, it should not be used as an excuse to continue to use formatted messages within new systems. Still

Table 4-2. Why Data Exchange Standards May Not Be Needed in 1995

Technology	Reason
Electronic Data Interchange (EDI)	A lack of multilevel security technology will prohibit connecting secure systems to civilian systems.
Graphics	The interchange of graphics will not be required (only the interchange of data from which graphics are derived).
Maps	Distributed on CD-ROM; no interchange required. The interchange of other geographical data will be required.
Video	Only required use may be videophone; standards which support other uses will not be needed.

much of the work that has gone into defining formatted messages, particularly efforts to coordinate data transfer between dissimilar systems, is useful because it helps standardization of data elements. The importance of standardizing common data elements cannot be overstated. The absence of such standards is a major impediment to interoperability regardless of the technology in the systems. Most data exchanged between CCIS nodes is exactly the kind addressed by these standards. At present, different nations, different Services, different systems use unique data definitions for common concepts, such as location or date-time group. Efforts to create a common standard are underway and are vital to future success. This section does not address such standards further.

Another need for formatted documents may come from electronic data interchange (EDI) applications. However, the relationship between systems that use EDI and the CCIS has not been explored, and CCIS requirements for EDI may not be clear for several years. Although there is no current stated need to integrate EDI into a CCIS in the target period, the technology to do so will be available. Two standards, ANSI X.12 and ISO/United Nations Edifact (ISO 9735) are in competition. Within the United States there is a thrust to abandon X.12 in favor of the international standard, but at least one powerful group, the automobile industry, has rejected a proposal to convert to Edifact by 1994. They are expected to migrate to Edifact, but at a much slower pace. Assuming other U. S. industries follow a similar tack, there may be a need to support both standards during the mid-1990s.

4.1.2 Unformatted Documents

Standards for unformatted documents are necessary not only for the electronic exchange of documents, but also as essential components of a group work architecture. The main issue, likely to be settled well before 1995, is exactly which standards will receive sufficient vendor backing. The issue is focused on the Office Document Architecture (ODA) and the Standard Generalized Markup Language (SGML), both ISO standards. SGML competes directly with Office Document Interchange Format (ODIF), an alternative encoding scheme for document structure. SGML and ODIF are both compatible with ODA. However, some experts believe that SGML combined with Document Style Segmentation and Specification Language (DSSSL) and Standard Page Description Language (SPDL) will be a much more comprehensive information management architecture than that of the combined ODA/ODIF standard.

4.1.2.1 Office Document Architecture/Office Document Interchange Format (ODA/ODIF)—ISO 8613

ODA is an architecture that enables transfer of the logical structure, content, and layout of office documents between applications, or from an application to various output devices. ODIF is an Abstract Syntax Notation (ASN.1) encoding of the document. The standard also defines the Office Document Language (ODL), an SGML encoding of the document. Even though ODA has been incorporated into both the NIST APP and the Computer-aided Acquisition and Logistics System (CALS), there is still disagreement on whether it belongs in GOSIP. Vendors claim to have implemented ODA using ODIF as an interchange format, but conformance testing criteria have not been proposed. Without evaluation information from several implementations, it is difficult to know if the standard is robust enough to handle all requirements. The connection between document logical structure, layout, and content is still an active topic of research. However, ODA is likely to have some use by 1995.

4.1.2.2 Standard Generalized Markup Language (SGML)—ISO 8879

SGML is a language for defining the logical structure of documents. Several implementations are available from vendors. It is primarily a framework and, as such, is not complete. While partial consensus on SGML has been reached, there is still disagreement on particular tag sets to be employed. (Tag sets are the common sets of document formatting codes

used in classes of document types. For example, technical manuals may use a tag set that is different from that used for management guideline documents due to differences in the audience, content, or custom.) The technology upon which SGML is based has existed for a long time in such document formatting products as Scribe, T_EX, and troff.

SGML is general to the extent that other representations and models can be included and represented within its framework. At the moment, it is not widely used. However, it is included in the NIST APP, which is likely to bring it into much wider use by 1995. The Graphics Communication Association (CGA) is discussing plans to produce a conformance test suite and a prototype test suite has been developed by the National Computer Systems Laboratory at NIST.

4.1.2.3 Document Style Segmentation and Specification Language (DSSSL)—ISO 10179

DSSSL defines specifications of document processing, such as formatting and data management functions, with the initial focus on formatting to both print and display media, and on data conversion. Future sections are expected to be added to cover the areas of data management. An objective of the DSSSL Standard is to provide a formal and rigorous means of expressing the range of document production specification, including high-quality typography required by the graphic arts industry. In addition, DSSSL includes General Language Transformation constructs which provide the capability to translate into an existing processing language, such as a text formatting language.

DSSSL is considered a companion standard to SGML in the following sense. SGML standardizes the representation of the document structure, while allowing users to develop their own techniques for interfacing with formatters. DSSSL provides a standardized architecture for the formatting specifications. It is expected that DSSSL will be used for interchange purposes, and that formatting programs will continue to use proprietary approaches to internal document representation.

DSSSL recently became a Draft International Standard, and implementations do not yet exist; it remains to be seen if vendors will support it as an interchange standard.

4.1.2.4 Standard Page Description Language (SPDL)

A Page Description Language is a language for communicating the content and structure of an image to an output producing device. Existing PDLs, such as Postscript, are proprietary. SPDL is an open standard for the same purpose. It recently became a Draft International Standard, and implementations do not yet exist; it remains to be seen if vendors will support it as an interchange standard.

4.1.3 Computer Graphics Metafile (CGM)—ISO 8632

In CGM, graphics data interchange is specified as a file format that is independent of device requirements and can be translated into the form needed by specific output devices, graphics systems, and computer systems. It is an ANSI and ISO standard and a FIPS. Vendors commonly use CGM as an exchange format for the storage, interchange, or output of a wide range of graphics (from slides for presentations to diagrams generated by scientific applications). Virtually all major microcomputer software products can generate or interpret CGM files. A CALS Application Profile that incorporates CGM has been proposed, and most CGM implementations conform to the CALS Profile. Three addenda are now being considered by ANSI and ISO. They add a global symbol capability, three-dimensional (3D) geometry, and improved engineering drawing. These changes will be upwardly compatible with the existing standard. Conformance test suites are available from several sources. A test service for conformance to the CALS Application Profile will begin in 1991.

4.1.4 Standard for the Exchange of Product Model Data (STEP)—ISO 10303

STEP is used in descriptions of engineered products that can be implemented on advanced manufacturing systems. Its role in a CCIS is in the dissemination of hardware maintenance information to field maintenance personnel. As with EDI, the availability of the technology is likely to precede the demand for its use in a CCIS, and the probability that it will be needed in the 1995-97 period is not high. The standard defines a complete product life cycle including all aspects of technical diagrams and documents in a neutral format for transmission over communications networks and processing by numerically controlled machine and assembly tools. This specification, currently in draft, is an extension to the ANSI Initial Graphics Exchange Specification (IGES). Extensions include the full life

cycle of products from initial requirements and design through final production and installation. Prototype implementations of small subsets are under way. STEP is still in draft and the majority of component specifications are projected to be ready in early 1992. The vendor community supports the standard, and use is expected to be extensive. Version 2.0 has been started but is not likely to be in use by 1995. If STEP fails to achieve adequate use by 1995, IGES may need to be considered. IGES and STEP are both in the proposed NIST APP. The CALS program is attempting to set the pace of technical data standards, and CCIS use should follow the CALS lead since the source of such data is likely to be CALS-related systems. (STEP was formerly known as the Product Data Exchange Specification, PDES.)

4.1.5 Maps and Geographic Information Standards

Non-defense use of maps and geographic information systems (GIS) is growing both in government and in industry. The most widely used source of domestic geographic information is the Department of Commerce's Topologically Integrated Geographic Encoding and Referencing (TIGER) system. The U. S. National Committee for Digital Cartographic Standards, a multiagency working group headed by the U. S. Geological Survey (USGS), is finalizing the Spatial Data Transfer Standard (SDTS), and the Department of Commerce is expected to convert to SDTS. Vendors are already anticipating converting their products to SDTS. In the meantime, the Defense Mapping Agency (DMA) has developed its own standard, the Vector Product Standard (VPS), which is incompatible with SDTS. In addition, the Digital Geographic Information Exchange Standard (DIGEST) is currently under development by the ten-nation Digital Geographical Information Working Group (DGIWG). Because of the immaturity of all these efforts, it is too early to know what subset of them will need to be supported in the 1995-97 time frame. It is possible that all will need to be supported because of the different requirements of the organizations involved.

4.1.5.1 Spatial Data Transfer Standard (SDTS)

SDTS, an object-oriented standard for encoding geographic objects, was originally conceived as a standard for transferring data between major producing organizations within the Federal Government, but is rapidly being accepted by GIS software vendors as an end user standard. However, the focus is on bulk data transfers on media such as magnetic tape or CD-ROM, rather than network exchanges from one computer to another.

4.1.5.2 Vector Product Standard (VPS)

VPS is DMA's effort to move away from raster images of printed maps to an encoding of geographic information in a form more usable by mapping software. After initially working within the SDTS project, DMA set out on what it considered a direction more oriented toward the end user instead of the major producer. VPS is structured along the lines of a relational database instead of the object orientation of SDTS. DMA still participates in the SDTS effort, and if SDTS becomes a FIPS, DMA is likely to take steps to bring VPS more in line with SDTS.

4.1.5.3 Digital Geographic Information Exchange Standard (DIGEST)

The Digital Geographic Information Working Group (DGIWG) intends to submit DIGEST to NATO for adoption as a STANAG (NATO Standardization Agreement). There has been some discussion within the DGIWG to submit DIGEST to ISO, but no plan has been put into place. Structurally, DIGEST is similar to SDTS.

4.1.6 Multimedia Standards

Multimedia technology is too new to have other than low-level standards. One important area, data compression, is covered below. Another area for which standards are rapidly converging is CD-ROM. Within a CCIS, there are likely to be significant applications for CD-ROM, but data exchange is not one of them. Still, CD-ROM standards are having some impact on other standards, particularly in audio, data compression, and multimedia file organization. Most efforts to derive standards in these areas are proprietary, controlled by a vendor or by consortia of a small number of vendors. The Multimedia Personal Computer (MPC) standard described below is one example of this activity. Its inclusion is not a recommendation to adopt MPC, but rather, a warning that if multimedia applications are desired in the 1995-1997 time frame, the only available standards may be proprietary ones.

4.1.6.1 Multimedia Personal Computer (MPC)

The initial MPC specification was announced last year by Microsoft as a multimedia standard for IBM PC-compatible computers. Since then, several vendors have joined with

Microsoft to announce supported products, and Microsoft has announced a certification program permitting computer vendors to advertise MPC-compliant systems. MPC is proprietary, but rapidly becoming a de facto standard in the PC compatible market. Several vendors sell systems, components, and titles for the MPC standard. No testing procedures exist. As yet, Microsoft has not announced plans to relinquish proprietary control of MPC nor to make it available on any other platforms.

4.1.7 Data Compression Standards

There are four standards listed in this group. While they compete in some areas, in general they have been designed for different applications. It is likely that all four will be viable standards for the 1995-97 frame. The degree to which each will be applicable to the CCIS will depend on the mix of applications. For example, the Motion Picture Experts Group (MPEG) standard will not be needed until digital motion video applications are built. Initial uses of MPEG will be in training materials and distributed on CD-ROM. Consequently, a CCIS built in 1995 may, in fact, incorporate MPEG as a data storage standard, but not as a data exchange standard.

4.1.7.1 Joint Photographic Experts Group (JPEG)—ISO 10918

JPEG was developed to support compression of full color still images for fax transmission. It is being adopted for storage and transmission of images independent of fax. Since it is a symmetrical compression algorithm, some vendors are also using it for full motion applications. Since JPEG is a "lossy" compression technique (that is, on decompression not all data is recovered), it is not suitable for applications where total fidelity with the original is required. The JPEG standard is based on the discrete cosine transform (DCT), the use of which for two-dimensional (2D) data compression is well understood. The ability to implement this technology in a single chip operating at real time speeds (30-60 frames per second) is new but does not require innovative technology. JPEG is expected to be approved as a standard in 1991, and chip- and board-level products based on draft versions of the standard are already in use. Vendors have had enough confidence in the standard to start shipping initial products since the spring of 1990. Since then, many vendors have incorporated JPEG in products being shipped and planned for release.

4.1.7.2 Motion Picture Experts Group (MPEG)—ISO 11172

MPEG is an asymmetrical data compression standard that combines DCT compression within a single frame with additional compression that eliminates redundancy between frames. Its intended use is with motion video, but since it is asymmetrical, it is very expensive to use in real-time applications, making it most suitable to publishing applications. MPEG is in draft, with full approval expected in 1992. Initial products should be available late in 1991. Single chip implementations that could have a significant impact on use have been promised before the end of 1992. MPEG is likely to be used heavily for applications where the images can be produced in a studio.

4.1.7.3 Joint Bilevel Imaging Group (JBIG)

JBIG is being developed for improving the quality of compressed images that have only two colors. It is expected to be used for compressing black and white photos and images of text documents, particularly in fax transmissions, since it produces superior quality images for the same bandwidth as Group 3 and 4 fax standards. JBIG is a recent development, still in early draft. To date, no products using JBIG are known to have been produced.

4.1.7.4 Video Coders for Audiovisual Services—CCITT H.261 and ANSI

The CCITT H.261 Recommendation (all CCITT standards are called Recommendations), *Video Coder for Audiovisual Services at p x 64 kbit/s (p x 64)*, was approved in late 1990. It describes a family of compression standards, one for every integer value of p , to provide a range of performance characteristics. Intended applications are for videophone ($p = 1$ or 2) and video conferencing ($p \geq 6$). A slightly modified version, *Digital Processing of Video Signals—Video Coder/Decoder for Audiovisual Services at 56 To 1,536 Kbits/s (P X 56)*, adopted by ANSI, is becoming a de facto standard for videophone and video conferencing applications. Interoperability between ANSI and CCITT-based systems is likely to be a problem. If videophone or videoconferencing with allies is desired, the participating U. S. CCIS may need to support both standards.

4.2 DATA MANAGEMENT SERVICES

In the 1995-97 time frame, standards and technology necessary to support data management will be starting to take shape. First generation products will provide much of the functionality needed to support data replication, distribution, and location transparency, but data model transparency is not likely to exist. Relational database management systems (DBMSs) will support binary large objects (BLOB), such as images and object-oriented data, but the BLOB and object-oriented interface definitions may not be standardized at that time. The standards described in the following sections are those most likely to be supported by vendors. Conforming implementations are likely to be limited to relational technology. There will be a need to support other data models, particularly object oriented, but implementations are likely to be proprietary until later in the decade.

Table 4-3. Data Management Standards
ISO 10032. Reference Model of Data Management.
ISO 9075.2. SQL-2.
ISO 10027. Information Resources Dictionary System (IRDS).
ISO 9579. Remote Database Access (RDA).
ISO 10026. Transaction Processing (TP).

4.2.1 Reference Model of Data Management—ISO 10032

The Reference Model of Data Management, like the more well-known OSI Reference Model, is not an interface standard. Rather, it describes a structure for how the parts of a data management system fit together. It describes a variety of alternative solutions to fit different problem situations. It is independent of data models, query languages, and distribution schemes. It defines a client-server relationship between various parts of the data management system, but does not specify any standards for communications between clients and servers. An annex to the standard indicates how existing standards fit into the structure, and identifies two particular areas in need of standardization efforts, both related to distributed data management. The first is a standard schema for distribution data and the services required for the definition and transfer of distribution data between systems. The second is a Remote Database Access (RDA) specialization to support communications for managing distributed data.

4.2.2 SQL-2—ISO 9075.2

Although most COTS relational database management systems are said to comply with the SQL-1 standard, few can interoperate and few applications are portable from one implementation of SQL to another. A goal of SQL-2 is to reduce the differences among implementations of SQL by tightening up the standard and making a part of it many of the features that are commonly proprietary extensions to today's systems. While this will not guarantee interoperability or portability per se, it should make it easier for users (in particular, application developers) to enforce interoperability and portability by limiting themselves to the standard parts of the language. The SQL-2 standard is likely to become final sometime in 1992, which should be more than adequate to ensure product availability by 1995, provided vendors are willing to allow standardization of features that currently differentiate their products from others.

4.2.3 Information Resources Dictionary System (IRDS)—ISO 10027

The purpose of an IRDS is to provide effective means for defining, manipulating, and updating data about information resources. IRDS's precursors have existed for many years in the form of data dictionaries, data directories, system catalogs, etc. However, these systems typically focused only on data stored in a single DBMS, not all data pertinent to an organization. IRDS is intended to address all data stored locally or remotely under the control of any number of DBMS or other systems. For command and control systems, it is critical that data be consistently defined and distributed appropriately (e.g., centrally, locally, federated), that the existence and location of data be known at each location where it is required, and that the data be maintained effectively. Appropriate use of an IRDS would aid each of these concerns.

ISO 10027 has not yet been approved as a standard, and a different version of IRDS has been adopted by ANSI and approved by NIST as a FIPS. Implementations of the ANSI IRDS specification have not yet been developed, and they may not be because the prevailing sentiment seems to be that the ISO version is far superior. Final approval of the ISO standard is expected in 1992, and it is expected to be the dominant standard by 1995. Convergence of the current ANSI and ISO IRDS efforts is extremely unlikely. Thus, it is likely that ANSI and ISO will have separate standards for IRDS in the near term. Both ANSI and ISO are exploring what the next generation standard IRDS will look like, but it will be at

least 5-10 years before a second generation IRDS standard is developed. Use of the ISO standard is recommended for the CCIS.

4.2.4 Remote Database Access (RDA)—ISO 9579

The goal of the Remote Database Access (RDA) effort is to allow the interconnection of applications and database systems from different manufacturers, under different managements, of different levels of complexity, and exploiting different technologies. This interconnection is achieved through the use of the Generic RDA Service and Protocol standard. The RDA standard defines a service facility that is provided to application programs which represents a boundary between the local processing of an application and that part concerned with communication. RDA may be used to carry database language commands and data between a client process and a database server to enable an application to read and update data in a remote database. Such commands must be established via "specialization" of the Generic Service and Protocol standard. A draft-proposed SQL Specialization for RDA has been developed.

The draft Generic RDA Service and Protocol standard is expected to be accepted as a standard in 1992. There is increasing interest in the RDA standards among vendors in the United States. Thus, it is likely that the commercial DBMS and protocol industry will produce RDA implementations by 1995.

4.2.5 Transaction Processing (TP)—ISO 10026

TP is the set of application layer services of OSI that support distributed transaction processing. Strictly speaking, as part of the application layer, TP should be described within the network portion of this report. However, the presumption is that the receiving end of a transaction service request is, in fact, a data management system. ISO 10026 provides a traditional two-phase commit procedure for transaction commitment. It is, therefore, expected to be adopted by the vendor community once it becomes an approved standard.

4.3 NETWORK SERVICES

Network services provide the communications necessary to implement applications dependent on distributed processing or a distributed computing environment. They provide

connectivity with flexibility and extensibility. These services are structured such that they can incorporate new or evolving technology, expand to include new nodes or subnetworks, and provide the flexibility to address CCIS specific problems (e.g., mobile nodes).

Table 4-4. Network Services
ISO 7498. ISO OSI Reference Model.
FIPS 146. Government Open Systems Interconnection Profile (GOSIP):
CCITT X.500, ISO 9594. Directory Services.
ISO 8671. File Transfer, Access, and Management (FTAM).
CCITT X.400 Series. Message Handling System (MHS).
Network Management:
ISO 10040 and 10165. Management Information Base (MIB).
ISO 9595 and ISO 9596. Information Processing Systems - OSI Common Management Information Service (CMIS) and Protocol (CMIP).
ISO 9040 and ISO 9041. Virtual Terminal (VT).
ECMA 127. Remote Procedure Call (RPC).
Telematic Standards:
Teletex. CCITT F.200, X.60, X.61, S.70, and X.430.
Textfax. CCITT T.6, T.61, T.72, and T.73.
Telefax. CCITT T.5, T.6, and T.73.

4.3.1 ISO OSI Reference Model—ISO 7498

The International Organization for Standardization Open Systems Interconnection (ISO OSI) Reference Model separates communications protocols and services into seven layers. This layered model has been adopted as the architectural model for network services. The seven layers, described in more detail in Appendix C, are Application, Presentation, Session, Transport, Network, Data Link, and Physical. ISO 7498, the international standard which specifies the reference model, is discussed in [Nash 1991]. Individual protocols and services are tailored to specific sets of functionality. Mission-specific applications can maximize their use of the communication system, and the benefits of network services, by selecting the protocols and services which most closely match their unique requirements.

In this section, the focus is on the Application Layer, since this is the layer visible to applications and users. (The reader is referred to [Nash 1991] for descriptions of services and protocols in Layers 1-6.) GOSIP, briefly described in the following subsection, is used

as the starting point for recommendations on network services standards for the WAM target profile. Accordingly, the Application Layer services and protocols that are expected to be part of GOSIP by 1995-97 are identified and described. Two Application Layer services (remote procedure call and telematic services) that at this time have not been added to the future GOSIP agenda are also described. It is recommended that these be considered for inclusion in the WAM target profile.

In addition to Application Layer services and protocols, the topic of network management is addressed in this section. Network security is discussed in Section 4.6. Both network management and network security are cross-cutting issues that impact multiple layers of the ISO OSI reference model.

4.3.2 Government Open Systems Interconnection Profile (GOSIP)

GOSIP addresses the need of the Federal Government to move immediately to multi-vendor interconnectivity without sacrificing essential functionality already implemented in critical networking systems. In other words, GOSIP is a set of rules for the specification of the interoperable capabilities of new ADP equipment to be procured by the Federal Government. Refinements of OSI protocols produced by ongoing OSI Implementors Workshops sponsored by NIST provide a basis for GOSIP. The workshops are composed of commercial vendors and users, which meet quarterly to provide a mechanism to keep GOSIP current.

The motivation for GOSIP's use of the OSI standards is an attempt to prevent further proliferation of different network protocols within the government. Since all new systems must support the OSI standard, the goal is to move to only one network standard, the OSI.

The ISO OSI standards will provide a common meeting ground for many different vendor's equipment, thus reducing the proliferation of private data network domains with each using their own, mutually incompatible, standards. Standardized hardware will also reduce the cost of data networks as well as make the upkeep process easier, since staff will have fewer different network "standards" to master.

GOSIP (Version 1) became a Federal Information Processing Standard (FIPS 146) in August 1988. GOSIP is to be used by all Federal Government agencies when procuring computer network products and services and communication systems or services that provide equivalent functionality to the protocol profile defined in the GOSIP documents. Ver-

sion 2 was published in October 1990, and Versions 3 and 4 are expected to be published by the 1995-97 time frame of the target profile.

At the application layer, Version 1 of the GOSIP document includes Message Handling System (MHS) and File Transfer, Access, and Management (FTAM). Version 2 includes the following additions: VT and ODA/ODIF (ODA/ODIF, being standards for data exchange, are discussed in Section 4.1). Version 3 will include directory services, extensions to VT, 1988 extensions for MHS and FTAM, and additional data exchange standards (e.g., CGM, SGML, and EDI, which are described in Section 4.1). Version 4 is expected to include standards in the area of data management services (e.g., TP and RDA, which are discussed in Section 4.2). It is expected that network management will be introduced into GOSIP in Version 3, and further developed in Version 4.

4.3.3 Directory Services—CCITT X.500, ISO 9594

Directory Services functionality is based on the CCITT X.500 and ISO 9594 standards. The directory is a hierarchical organization of objects or records. Each object is a set of attributes and associated values. The allowable values, for any attribute, can be restricted to a particular data type or range of values. Directory Services are used to retrieve, set, and search for attributes or their values. This basic functionality can support a wide range of applications.

For example, a system may consist of a large number of nodes and an even larger number of users. It is unreasonable to expect all CCIS users to remember all relevant information about other CCIS users. A CCIS user may request that a message be sent to a particular point of contact. The MHS can augment point of contact information with all the data necessary to deliver the message. For example, a point of contact could be an office or organization. The MHS could search, using Directory Services, the directory for a contact record consisting of point of contact and person to contact. It replaces the point of contact with this person and performs another search of network information records. It retrieves the network address and network routing information, from the record. This information allows the MHS to actually send the message to the person to contact.

4.3.4 File Transfer, Access, and Management (FTAM)—ISO 8571

The services and protocols associated with FTAM are described in ISO 8571. These protocols and services are used to organize or manage, access, and transfer files. FTAM provides basic functionality for handling data. Data is collected into files. Files can be accessed (opened, read, written, and closed) from the local host or CCIS node and transferred to any other host or CCIS node. A typical file organization is hierarchical. In a hierarchy, files can have parents (referred to as directories) and siblings. FTAM can be used to create and delete files or directories within this hierarchy.

The data within a file can be of any form. A text file could hold a message, electronic mail, analysis, or a complete report. The results of a simulation or model could be stored in a data file. Other files could contain graphics, imagery, photos, and maps.

4.3.5 Message Handling System (MHS)—CCITT X.400

The MHS provides the services and protocols to communicate electronic mail and messages. This service does not restrict messages to “text only” formats. The standards address multipart documents. These documents can be multimedia. They can include text, audio, and video sections. The relevant MHS standards are the CCITT X.400 series recommendations.

4.3.6 Network Management

Network Management provides the facilities to describe, control, and manipulate the operational aspects of the communications network. The actual description and the indirect manipulation and control of the network occurs through the Management Information Base (MIB). The data in the MIB can describe all aspects of the communications network. This includes Accounting, Configuration, Fault, Performance, and Security Management Information. The organization of the MIB, attributes, and naming conventions are addressed in ISO standards 10040 and 10165. Interaction with the MIB is through the services and protocols described in ISO 9595 and ISO 9596.

The Network Management services and protocols are currently able to address any communications network. The organization of the MIB, attributes, and naming conventions

are still subject to change. Advances in communications technology may require new management information attributes, objects, or descriptions.

4.3.7 Virtual Terminal (VT)—ISO 9040, 9041

Virtual Terminal (VT), described in ISO 9040 and ISO 9041, defines the protocols and services which are sufficient to address the common features of full feature character terminals. VT is defined for the ISO 8859/1 character set. It understands the difference between command or control characters and text characters. Text characters are displayed. Control characters result in text manipulation or character positioning. Text manipulation includes the page operations and scrolling. Character positioning or control includes line feed, carriage return, backspace, and horizontal and vertical tabs.

4.3.8 Remote Procedure Call (RPC)

An RPC-based application replaces subroutine calls with RPC calls. Subroutine calls perform processing on the local machine and within the same process. RPC calls can invoke processing on the local machine or a remote machine. RPC is not (currently) scheduled to become a part of GOSIP. There are no ISO standards which describe it. ECMA 127 is the only international standard which addresses it. However, client-server based computing is becoming an accepted technology and RPC is one method to implement client-server computing. It is likely that RPC services and protocols will be standardized in the 1995-97 time frame.

4.3.9 Telematic (Teletex, Textfax, Telefax) Standards

Telematic Services standards describe the protocols and services applicable to devices designed specifically for correspondence and graphics or facsimile equipment. The services include Teletex, Textfax, and Telefax services. The Teletex service describes the protocols and services to be used for the interconnection of a correspondence-specific terminal to the open systems environment. The Telefax service describes the protocols and services to be used for the interconnection of a graphics or facsimile device to the open systems environment. The Textfax service describes the protocols and services applicable to devices which combine Teletex and Telefax services. The applicable standards are CCITT recommenda-

tions F.200, X.60, X.61, S.70, and X.430 for Teletex, CCITT recommendations T.6, T.61, T.72, and T.73 for Textfax, and CCITT recommendations T.5, T.6, and T.73 for Telefax.

Teletex, Textfax, and Telefax are not currently scheduled to become a part of GOSIP. In the 1995-1997 time frame, it is possible that evolving technologies, such as multimedia, will obsolete Telematic Services. However, it is also possible that a CCIS node will need to interoperate with an existing Telematic Services based network.

4.4 OPERATING SYSTEM SERVICES

POSIX is a suite of operating system interface standards being developed by IEEE Project 1003. It has a broad base of support within the Federal Government and within industry. Moreover, it is recommended in the WAM DCP [1989] as a component of the WAM OSE. For these reasons, POSIX was selected by IDA as the primary source of operating system interface standards for the WAM target profile early in the course of the investigation. POSIX standards define application program interfaces to underlying operating

Table 4-5. Operating System Services

IEEE Project 1003. Portable Operating System Interface for Computer Environments (POSIX):
IEEE 1003.1-1990 (ISO/IEC 9945-1:1990). Basic Operating System Services.
IEEE P1003.5. Ada Bindings.
IEEE P1003.6. Security Extensions.
IEEE P1003.2. Shell and Utilities.
IEEE P1003.7. System Administration.
Distributed System Services:
IEEE P1003.8. Transparent File Access.
IEEE P1003.12. Protocol Independent Interface.
IEEE P1003.17. Directory Services API.
IEEE P1224. X.400 Mail Services API.
IEEE P1238. Common OSI API and File Transfer, Access, and Management (FTAM).

system services. In doing so, they define the functionality of the operating system services to the extent that the functionality is visible at the application program interface level. POSIX standards do not define the internal structure or implementation of underlying operating systems.

POSIX standards are based on Unix. While Unix is some sense a de facto standard, it has many variants, which impede application portability. The POSIX standards represent an attempt to standardize Unix interfaces for the purposes of application portability at the source code level. The first POSIX standard, POSIX.1 (developed by IEEE working group 1003.1), defines the interfaces to basic operating system services. In the Unix tradition, it is oriented toward the interactive, multi-tasking, multi-user application domain. Using POSIX.1 as a baseline, other 1003 working groups are extending POSIX to additional application domains. Again, the POSIX standards define interfaces, not implementations; while POSIX does specify Unix-style interfaces, it does not mandate a traditional Unix-style implementation. In Appendix D, requirements for CCIS operating systems are enumerated. The requirements dictate how CCIS operating systems must manage resources and what types of interfaces they must provide. The operating system requirements are derived from several high-level CCIS requirements:

- a. A CCIS will provide an interactive, multi-tasking, multi-user environment.
- b. A CCIS will provide a distributed environment.
- c. The maximum degree of fault tolerance will be provided by the system, in accordance with the criticality of the application.
- d. There will be a real-time system or subsystem capability.
- e. The system will provide adequate security to protect sensitive and classified information in accordance with relevant directives and laws.
- f. A CCIS will provide system administration to aid all other requirements.
- g. The Ada programming language will be used to the maximum desired degree for developing systems and applications software.

Simply stated, the operating system requirements are to support these high-level CCIS requirements. With some exceptions, POSIX can support these requirements. The exceptions are in areas of active research, such as security, fault tolerance, and system management, where the consensus necessary for standardization has not been reached and open system standards have not been developed. Because POSIX has a broad base of support, it is anticipated that as consensus in these areas develops, standards embodying the consensus will be introduced. In the meantime, the use of vendor-proprietary or application-specific approaches to these areas will have to continue.

In the following subsections, the component standards of the POSIX suite that are recommended for inclusion in the WAM target profile are identified and briefly described. The

description of each component standard includes a statement of which high-level CCIS requirements are addressed by the standard.

4.4.1 Basic Operating System Services—IEEE Standard 1003.1-1990

IEEE Standard 1003.1-1990 (ISO/IEC 9945-1:1990), informally referred to as POSIX.1, addresses the requirement for interactive, multi-tasking, multi-user environments. It defines application program interfaces to basic operating system services. The interfaces cover processes, process environments, signals, timer operations, files and directories, pipes, file I/O, and terminal device management.

POSIX.1 has two limitations: it is oriented toward centralized computer systems and toward the C language. Both are being addressed within the Project 1003. Application programming interfaces to standard distributed services are being developed by several 1003 working groups. The distributed services enable communication and interoperation among multiple POSIX-conforming systems (and, in fact, any systems that adhere to the standard protocols, such as FTAM, that underlie the distributed services). In addition, as 1003 working group members define new interfaces, they are careful to ensure that the interfaces apply to POSIX-conforming operating systems running on parallel or distributed processors. In this latter case, the POSIX-conforming operating system would be implemented as a distributed operating system. That is, it would manage resources and provide interfaces in such a way that interconnected processors are transformed into a single logical system. With respect to programming languages, a language-independent specification is being developed, as discussed in the following subsection.

4.4.2 Ada Bindings—IEEE P1003.5

The P1003.5 standards address the requirement for Ada. Not only POSIX.1 but also almost all of the other POSIX standards have a C language bias, due to historical ties and a continuing close relationship between Unix and C. An effort is being made to remove the C language dependencies from the POSIX family base, driven in part by the desire to continue to carry POSIX into the international standards arena. The current plan is to develop language-independent specifications for all POSIX interfaces, and then to supplement the language-independent specifications with language bindings.

The 1003.5 working group is defining Ada language bindings for POSIX standards. The first product of the working group is an Ada binding to POSIX.1, which is undergoing the IEEE ballot resolution process. In July 1991, the working group began work on Ada bindings to the POSIX real-time extensions being developed by the 1003.4 working group.

4.4.3 Real-Time Extensions—IEEE P1003.4

The P1003.4 standards address the requirement for real-time responsiveness, and, to a lesser extent, the requirements for distributed environments and Ada. In addition, they support parallel computing, which is important for high-performance applications.

The P1003.4 standards address issues that are of particular concern to real-time applications developers, who generally consider a traditional Unix environment to be unacceptable for fielding their applications. In real-time systems, resources must be managed so that time-critical application functions can control their response time, possibly resulting in delay or even starvation for non-time-critical application functions. Therefore, the 1003.4 Working Group has focused its initial efforts on defining application interfaces to the functional areas that impact resource management, for example, priority scheduling, real-time files, and process memory locking.

The P1003.4 standards provide support for distributed and parallel computing through their interprocess communication and synchronization interfaces. These interfaces are designed to apply to any processes running in a POSIX system, whether the system consists of one processor or several processors.

The P1003.4 standards provide support for Ada and for parallel computing through the POSIX *thread* mechanism proposed in P1003.4a. The POSIX thread is a lightweight process (i.e., a thread of control with minimal execution context associated with it); multiple threads can concurrently execute within a POSIX process (which provides the address space and other execution context, such as open file descriptors, which are shared by the threads existing within the process). Thus, the thread can be used in real-time programs, Ada programs, or parallel programs running on shared memory multiprocessors as a fast, efficient unit of concurrency within a shared address space.

4.4.4 Security Extensions—IEEE P1003.6

The P1003.6 standard addresses the requirement for security. The P1003.6 standard defines interfaces to security services and mechanisms. The working group's basis for consideration of security issues is DoD 5200.28-STD, *Trusted Computer Security Evaluation Criteria* (TCSEC or the "Orange Book"). Currently, the major features covered by the standard include discretionary and mandatory access control, audit, privilege, and information labeling. It is important to note that the P1003.6 standard defines interfaces that enable these features to be used; the standard does not address the underlying implementation, which clearly determines whether or not an operating system is trusted to ensure security. The implementation requirements are covered in the Orange Book. The purpose of P1003.6 is to enhance application portability across implementations of secure operating systems. As is true of the Orange Book, the focus of the P1003.6 standard is confidentiality. Other aspects of security, such as integrity and availability, are not covered.

4.4.5 Shell and Utilities—IEEE P1003.2

The P1003.2 standards primarily address the requirements for interactive and multi-tasking/multi-user environments. They complement POSIX.1. While POSIX.1 focuses on basic system services, the P1003.2 standards focus on another distinctive and popular aspect of Unix—its shell and utilities. The shell is a command language interpreter. The P1003.2 standard, *Information Technology—Portable Operating System Interfaces (POSIX)—Part 2: Shell and Utilities*, defines the shell command language. It also specifies standard interfaces for many common Unix utilities. Its purpose is to aid application portability. The P1003.2a standard, *User Portability Extension*, covers additional terminal-oriented utilities, including a full-screen editor.

4.4.6 System Administration—IEEE P1003.7

The P1003.7 standard addresses the requirement for system administration. It covers topics such as backup, recovery, system startup, system shutdown, clocks, print spooling, and file management.

4.4.7 Distributed System Services

Several POSIX working groups are addressing the requirement for distributed environments. They are defining application program interfaces (API) for standard computer network protocols and services. Until recently, the network standards community has focused on the goal of interoperability of heterogeneous computer systems and application software running on the systems. The approach has been to define standard peer-to-peer protocols. Application portability across different implementations of the protocols has not been of concern. Project 1003 has begun to address this issue, by undertaking the development of a standard API for each of several network services, including the following:

- a. 1003.8, Transparent File Access. This group is defining an API for a transparent file access facility similar in functionality to Sun Network File System (NFS).
- b. 1003.12, Protocol Independent Interface. This group is defining an API for a data transport capability that is not tied to any particular network protocol.
- c. 1003.17, Directory Services API. This standard will be based on CCITT Recommendation X.500.
- d. 1224, X.400 Mail Services API. This group plans to define an API for a mail service based on CCITT Recommendation X.400.
- e. 1238, Common OSI API and File Transfer, Access, and Management (FTAM) API. This group is defining an API for FTAM (ISO 8571) and other OSI application services.
- f. Remote Procedure Call (RPC) API. ANSI X3T5.5 is defining an API for a remote procedure call facility, which enables procedure calls to be made across a data communication network. The IEEE has decided to support the ANSI effort, rather than to embark on an independent course.

4.5 PROGRAMMING SERVICES

Programming services affect the development and execution of applications. In the former case, programming services support the development, checkout, installation, maintenance, and testing of application and system software. At present the only standards supporting this domain are programming language standards; moreover, from this set of standards, only those for Ada and C are recommended for inclusion in the WAM target pro-

file. However, work on Ada 9X, C++, and other languages, as well as on software engineering environments, and program development and maintenance tool interfaces and data models, is moving toward the stage where standardization is likely by 1997.

Table 4-6. Programming Services

ANSI/MIL-STD-1815A, ISO 8652, Ada.
Ada 9X.
ANSI X3.159-1989, C Programming Language.
C++ Programming Language.
Other Languages: Prolog and Common LISP, ANSI committee X3J13.
MIL-STD-1838A, CAIS-A.
ECMA PCTE+.

4.5.1 Ada—ANSI/MIL-STD-1815A, ISO 8652, and Ada 9X

Ada was developed in the late 1970s to give the DoD a single, common high-order programming language with which to build reliable, portable, and more cost-effective software. Ada is an ANSI, ISO, and military standard. The current standard was approved in 1983. The use of Ada is mandated for CCIS software and therefore will be used extensively in the WAM program. Although a rigorous set of tests are used to check Ada compiler conformity with the standard, many Ada compilers sacrifice performance for conformity. This has led to some criticism of the ability of Ada to meet its mission.

The Ada 9X program is currently underway to revise ANSI/MIL-STD-18151A. The focus is to clear up inconsistencies with the semantics of the current standard and to refine certain features to better address the applications to which Ada is applied. Ada is mandated by the DoD so its use is required, but the revision of Ada is expected to make it more accepted even if there were no mandate. Developing new systems or re-engineering old systems into Ada will enhance portability and maintainability while promoting uniformity. Requirements for language revision have been incorporated in a Mapping Document which proposes the changes to be made to sections of the Language Reference Manual (LRM) for Ada. A draft version of the Mapping Document was circulated for public comment in February 1991. Work is in progress to revise the content of the LRM which will be submitted for ANSI canvass in mid-1992 (June) with adoption of the revised Ada language standard.

targeted for March-April 1993. DoD, ISO and FIPS adoptions are expected in parallel with ANSI adoption.

4.5.2 C Programming Language—ANSI X3.159-1989

C is a general-purpose programming language designed for use in various types of software including operating systems and system level software. It is considered to be a high-level language, but its strength lies in the capacity to interact directly with the underlying system using built-in intrinsic operating system functions. The future CCIS will make use of COTS products as much as possible. Many COTS products are written in C, and some of these may be applicable to the future CCIS.

At present C is not formally tested for conformity to the ANSI standard, but commercial test suites for testing are available. Although the C standard is the result of refinements made since the mid-1970s, some instability is still expected, but no major update to the standard is planned.

4.5.3 C++ Programming Language

C++, designed for object-oriented programming and based on C, has been under development since the mid-1980s. While compilers currently exist, there is no formal standard to which they must conform. However, ANSI X3 is leading a standards effort for C++.

4.5.4 Other Languages

The development of expert system software for use as decision-making aids is promising. There are currently no standards for knowledge-base specification languages or notations. In the absence of such standards, the requirements for an architecture of the CCIS should be constructed so as not to preclude the use of expert systems in the future. A CCIS architecture that allows for such aids is desirable. Prolog and Common LISP are two languages associated with knowledge-based software. ANSI committee X3J13 is working on drafts for Common LISP and Prolog standards.

4.5.5 CAIS-A—MIL-STD-1838A

The Common Ada Programming Support Environment (APSE) Interface Set (CAIS) project was begun in 1982. The goal of the project was to ensure better development and maintenance environments for DoD mission-critical computer software. The CAIS is a set of interfaces through which APSE tools can access the operating system services provided in the host environment and communicate among themselves. Typically a tool that uses services provided by a given host is host dependent. Data or files generated by that tool will follow the conventions of the host. By standardizing a set of interfaces supporting a wide variety of hosts, tool portability is enhanced. This standard set of interfaces does not provide every operating system facility, just those most common and useful or necessary to ensure tool and tool database portability. In addition, the CAIS provides an object management system to ensure data integrity. The CAIS interfaces are specified as a set of Ada package specifications for services such as process control, file management, and device control. CAIS-A became a U. S. military standard, MIL-STD-1838A, in 1989.

4.5.6 ECMA PCTE+

The European Computer Manufacturers Association (ECMA) Portable Common Tool Environment-Plus (PCTE+) project was begun in 1983 by the European Strategic Programme for Research in Information Technology (ESPRIT) [Thomas 1989]. The goal of the PCTE+ project was to describe and build prototypes of tool interfaces that could be used to define a highly secure software development environment. The environment would include a set of public tool interfaces and a data management system. Tool builders might use the interfaces to either integrate or attach their tool products to an environment. The services provided by PCTE+ include data management, tool execution and communication, distribution and environment management, and programmer interface for user interface management. The PCTE+ standard is still under development but is expected to receive ISO approval by 1995.

4.6 SECURITY SERVICES

The requirements for CCIS security services include data confidentiality, data and system integrity, and system availability protection. The ability to protect for confidentiality

exists in several operational modes (e.g., system high, dedicated, controlled) and is evolving in robustness in the multilevel mode. The ability to protect data and system integrity is dependent upon hardware, system software, application software, and organizational policy. Such protection may exist to a limited degree, but there are few criteria for evaluating its effectiveness. The situation with regard to system availability is similar. Basic and applied research is needed to develop criteria and standards that will cover integrity and availability. It is doubtful that a significant array of standard products to fulfill all three aspects of protection will exist during this decade or even the early part of the next decade. Only confidentiality is discussed further in this section.

Table 4-7. Security Services	
DoD "Rainbow" Series of documents.	
CSC-STD-003-85. Computer Security Requirements ("Yellow Books").	
DoD 5200.28-STD, Trusted Computer System Evaluation Criteria (TCSEC) ("Orange Book").	
NCSC-TG-005 Version-1. Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria (TNI) ("Red Book").	
NCSC-TG-021 Version-1. Trusted Database Management System Interpretation of the Trusted Computer System Evaluation Criteria (TDI).	
Secure Data Network System (SDNS) Standards.	
IEEE STD 802.10. Standard for Interoperable LAN Security (SILS).	

The confidentiality mode requirement for CCIS security services is multilevel secure operations. That is, the system, operating with concurrent access by users cleared to different levels of trust, will be responsible for maintaining the separation of compartmented and hierarchically classified and unclassified data, and controlling access to and flow of that data in accordance with specified security policy.

Security services pervade the architecture, and elements are located within all of the other services. However, security standards and products that enable multilevel operations are not yet mature enough to specify with confidence a complete and explicit structuring of the security services underlying the CCIS. In lieu of such a specification, an evolutionary strategy is offered that transitions current secure operational modes while placing transitional elements into the architecture over time to move ultimately into multilevel secure operations.

Maintaining the mode of operations of a CCIS at "system high" will have minimal impact on the architectural standards, but may have immediate impact on the CCIS archi-

ecture itself. Users will be cleared to the highest levels of information sensitivity and the data will be allowed to coexist without explicit separation within the system. Need-to-know will be the basis of access control. A Class C2 system (discretionary security) is already required by the DoD Computer Security Requirements [CSC 1985a] as classified information is being processed and individual accountability must be accomplished. The immediate impact occurs where a current CCIS has not moved to Class C2.

The next appropriate evolutionary alternative to system high might be dedicated subsystems, e.g., one subsystem (operated at system high: TOP SECRET plus compartments) dedicated to intelligence or specifically compartmented information and another subsystem dedicated to the remaining C² operations (system high: SECRET). Such an approach offers an incremental solution for reducing the number and types of clearances required. Users have both the clearance and the need-to-know for all the information in their dedicated subsystem. However, should there be multiple compartments and all users either not cleared or not having the need-to-know for all compartments, a mandatory multilevel secure system would be required (e.g., Class B2, because of the ranges of classification levels in the dedicated subsystems). Assuming that a mandatory multilevel secure system is not required (because users have the clearances and need-to-know), dedicated subsystems may have minimal impact on architectural standards, but may significantly effect the architecture and system operations in terms of cost and unique or duplicated components. Each dedicated subsystem will require a C2 system since individual accountability is required whenever classified information is being processed. From this dedicated configuration, a less complex migration to multilevel operations using, perhaps, class B2 systems would be possible.

Multilevel security, however, significantly impacts every architectural standard, the architecture itself, and, depending on the acquisition strategy, overall system costs as well as the operations of the system. Multilevel security is the proverbial "long pole" in the tent. With respect to near-term CCIS implementations, such as the WAM program, a policy for incrementally implementing multilevel security is essential as there remains a lack of agreed-upon security standards and implemented technology embodying those standards. As standards bodies progress on multilevel and military-specific security issues and as technology conforms to those standards, a future CCIS will have less of a problem in specifying and implementing their security architectures. Many efforts are underway to advance these standards and expectations are that some standards should be available in the 1995-97 time frame. However, products based upon these standards may not be available for several years after the standard, especially if product evaluations and certifications are

required. In the interim, we may experience more ad hoc security solutions. The goal is to reduce the number of such ad hoc solutions.

In achieving multilevel secure operations the architectural philosophy, supported by system-high mode of operations or by dedicated subsystems, should concentrate first on incrementally evolving multilevel network security via data encryption, physical separation, and trusted gateways and guards, and then second, on addressing the multilevel issues of the other services supporting C³I applications, (e.g., Operating Systems, Data Management, User Interface). A brief rationale for this approach is discussed below.

- a. Data encryption enables sharing of transmission facilities as well as user, device, and message authentication, access control, confidentiality, non-repudiation, and encapsulation for integrity protection. Data encryption technology provides hardware and software components that permit enforcement of a network security policy to be incorporated into standard network protocols. Encryption key distribution and management is required. Data encryption technology is generally available and specific "open" standards are evolving. End-to-end data encryption is not expected to significantly impact the operational missions of the CCIS. System-high mode operations eliminate the need for mandatory multilevel protection of the hosts and gateways within the bounds of the secure network.
- b. Physical separation enables isolation of disparate users, systems, or data where encryption is not available or cost effective. Isolation may significantly impact mission operations as well as costs.
- c. Trusted gateways and guards provide access filtering and mediation for disparate users, systems, and data. These gateways and guards will require technology trusted to handle mandatory multilevel secure operations (e.g., Class B2). This technology is currently available, but from a limited number of suppliers and the technology is not necessarily oriented toward open systems.
- d. Use of encryption and dedicated subsystems may allow approaches to gateways and guards that reduce the multilevel trust requirements such that a much wider range of trusted technology products can be considered (e.g., Class C2). The performance capabilities of this trusted technology may significantly impact the operational mission and the products themselves may significantly impact the cost of the architecture.

- e. The final longer-term evolutionary aspect is incorporating multilevel security into all the other service areas. By using the strategy outlined above, a relatively smooth transition (compared with a more abrupt approach) to multilevel secure mode can occur, but more important, this approach may lessen the likelihood of loss of security.

Standardization initiatives to address security within the ANSI and ISO standards bodies have begun. Earlier standards for security and other services, inadequate in specific military aspects of security, are being evolved to address issues (e.g., access control lists, labels). The majority of security standards within the ISO purview remain working drafts or incomplete working drafts as shown on the Status of Security Coordination within SC21 [SC21 1991]. There is some concern being raised that the coordination of security work items among the various working groups within ISO is not sufficiently effective and that gaps and duplicative work are occurring with increasing regularity. With respect to ANSI standards, the DDI (Director, Defense Information) has stated his intent to reinvigorate the DoD to push for ANSI standards, as ISO standards are taking too long to produce. This push, when implemented, may significantly influence the security standards and their availability for the security services profile.

Among the most important security standards from a CCIS perspective are the DoD standards that are having an impact on commercial industry as well as ANSI or OSI standards, and the industry standards that can potentially meet military security needs. The DoD "Rainbow" Series of documents provides criteria for trusted technology development (evaluation) as well as criteria for what level of trusted technology is appropriate for a particular environment. These documents are listed below:

- a. CSC-STD-003-85, *Computer Security Requirements*, also known as the Environments Guideline and more popularly as one of the "Yellow Books," is key to establishing what class of trusted technology product is needed for a particular environment, the starting point for developing the security services to be placed in the CCIS architecture.
- b. DoD 5200.28-STD, *Trusted Computer System Evaluation Criteria* (TCSEC), also popularly known as the "Orange Book" is a path-setting document that has been the driver for the current array of trust technology products. There are indications that finer-grained criteria from Europe and the development of integrity criteria by NIST may slightly delay the commercial influence of this document.

- c. NCSC-TG-005 Version-1, *Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria (TNI)*, also popularly known as the "Red Book." This document has only had limited use, primarily to evaluate local area networks. As more distributed system security issues arise, this document will exert more influence on the trusted technology products to be used.
- d. NCSC-TG-021 Version-1, *Trusted Database Management System Interpretation of the Trusted Computer System Evaluation Criteria (TDI)* extends evaluation criteria to trusted applications in general and database management systems in particular. Published in April 1991 for a one-year trial to gather experience on clarity, technical accuracy, and utility, its effect on the trusted technology is beginning to gather momentum.

While BLACKER and KG-84's may provide the more immediate (1995) end-to-end and link encryption approaches to network security, the Secure Data Network System (SDNS) Standards [1989a-k], once implemented, are considered more appropriate for inclusion in the target profile due to the expected overall capabilities and closer alignment of SDNS with open systems standards. The goals of SDNS are to create specifications for end-to-end security; to utilize the OSI Reference Model; to design an architecture to include electronic mail and end-to-end encryption; to provide transparent key management; and to demonstrate feasibility of techniques. The National Security Agency (NSA) is supporting the SDNS project; several standards for security protocols have been released to the public domain. NSA is working with NIST to incorporate the SDNS protocols into GOSIP. The SDNS protocols will also be introduced into ANSI by NIST and, if accepted, into the ISO OSI Security Architecture. SP3 (Secure Protocol for Layer 3) and SP4 (Secure Protocol for Layer 4) have already been submitted by ANSI to ISO; SP4 has been accepted as a new work item, and SP3 is expected to be accepted as a new work item after some modifications. Testing of breadboard hardware with the SDNS protocols was conducted in 1989.

IEEE STD 802.10, Standard for Interoperable LAN Security (SILS), is an industry standard that provides a security protocol at layer 2. It consists of four parts: Part A: The Model; Part B: Secure Data Exchange; Part C: Key Management; and Part D: Security Management. Parts A, C, and D are still in development, Part B is in the IEEE 802 approval process and is independent of the other parts. Part B contains the description of the Secure Data Exchange Protocols which provide services of data confidentiality, connectionless integrity, data origin authentication, and access control. Part B of this standard was submitted by ANSI to ISO/IEC JTC1/SC6 for consideration by SC6 at its 8-9 July 1991 meeting.

Desired consideration included review and comment and support for a SC6 New Project (NP) on LAN security.

4.7 USER INTERFACE STANDARDS

Standard user interfaces will ensure a high degree of application portability and provide a consistent way for developers, administrators, and users to gain access to application programs. The standards recommended for use in the target profile are outlined below. More detail about the content and status of these standards can be found in [Nash 1991]. Appendix G provides more detail on their use in a CCIS user interface.

Table 4-8. User Interface Standards	
User Interface Architecture:	
User Interface Services Reference Model (UISRM).	
X-Windows (or X).	
Toolkit and User Interface Management System (UIMS) Standardization:	
OSF/Motif.	
OpenLook. Sun Microsystems and AT&T.	
XVT (eXtensible Virtual Toolkit).	
Graphics Services:	
Graphical Kernel System (GKS). ISO 7942, ISO 8651. GKS-3D, ISO 8805, 8806.	
Programmer's Hierarchical Interactive Graphics (PHIGS) Interface. ISO 9592, ISO 9593.	
Computer Graphics Interfacing (CGI). ISO 9636.	
Color Standards:	
1976 CIE Uniform Chromaticity Scale (CIE-UCS).	
Color Standard: CIE L*, u*, v* (CIELUV).	
Rendering Standardization: Pixar RenderMan.	
Human Factors Standards:	
MIL-STD-1472. Human Engineering Design Criteria for Military Systems, Equipment, and Facilities.	
MIL-H-46855B. Human Engineering Requirements for Military Systems, Equipment, and Facilities.	
MIL-HDBK-761A. Human Engineering Guidelines for Management Information Systems.	
MIL-HDBK-763. Human Engineering Procedures Guide.	
ESD-TR-86-278. Guidelines for Designing User Interface Software.	

4.7.1 User Interface Architecture

In order to meet CCIS portability and interoperability requirements, the user interface target profile is based on the proposed NIST APP. The user interface service specifications are based on the User Interface Services Reference Model (UISRM) [Kuhn 1990], a layered conceptual model. The layers are Data Stream Encoding, Data Stream Interface, Subroutine Foundation, Tool Kit Components, Presentation, Dialogue, and Application. The approaches supporting the APP proposed for the target profile are discussed below.

4.7.1.1 Window Manager Standardization

A window management system provides for the creation, manipulation, and deletion of windows on a graphics display device. The major goals of a window management system for the CCIS user interface include high-performance with high-quality text and graphics; network transparency; support for many different applications and management interfaces with true multitasking of applications; portability and device-independence for both applications and the window management system; and configurability and extensibility of the window management system.

X-Windows (or X), a window management system developed at MIT, is the focus of ANSI and IEEE standardization efforts, a standard part of Unix System V, and part of the operating system to be released by the Open Software Foundation (OSF). The wide use of X means that it currently provides the best basis for interoperability at the windowing level. It defines a C language source code level interface to a network-based bitmapped graphic system. It allows devices to be shared among several processes at the same time and allows access to devices from remote sites. The FIPS standard (FIPS 158, X-Window User Interface) provides specifications for the APP Data Stream Encoding, Data Stream Interface, Subroutine Foundation, and Tool Kit Components layers.

4.7.1.2 Toolkit and User Interface Management System (UIMS) Standardization

A UIMS handles all input aspects, including all visible parts of a display and all aspects of dialogue between the user and application. From the CCIS point of view, the primary advantages offered by a UIMS are the separation it provides between user interface code and an application, and the provision of a mechanism for abstract specification of the user

interface syntax and semantics. In the absence of accepted standards there are two main industry thrusts to consider: OSF/Motif and OpenLook. Both are based on X. OpenLook, jointly developed by Sun Microsystems and AT&T, is a device-independent UIMS that supports multitasking graphics user interfaces. The user model is essentially object oriented and is intended to support a wide range of users. Motif is a hardware, network, and operating system independent product that incorporates Digital Equipment Corporation's User Interface Language for describing visual aspects of the user interface, Microsoft Corporation's Presentation Manager behavior for user skills portability, and Hewlett-Packard's 3D appearance. It conforms with the X/Open Consortium's Portability Guide to support both 16-bit and compound strings for Asian and European languages.

NIST has selected XVT (eXtensible Virtual Toolkit) for APP Presentation and Dialogue layers. XVT has also been selected by the IEEE P1201.1 Standards Committee as the basis for its standard application program interface for portable graphical user interface applications. Based on X, it determines appearance of the user interface specifying, for example, how the components provided in the Toolkit Components layer should be composed to create windows. XVT includes its own virtual toolkit to allow mapping to various UIMS, such as OpenLook and OSF/Motif. It includes a resource compiler that implements a portable resource specification language and an object-oriented, line drawing program written using XVT.

4.7.2 Graphics Services

CCIS requirements specifically call for graphics capabilities to support direct manipulation user interfaces, geographical information systems, 3D displays, and computer graphics conferencing.

4.7.2.1 Graphics Subroutine Library Standards

Graphics subroutine libraries support graphical information sent to and received from the screen and associated input devices. The basic device independence they offer can be increased by incorporating calls to a graphics package into a toolkit or UIMS, instead of embedding them in the application code. Standards that provide a textual specification for graphics functions are supported by an implementation of the commands. Language bindings provide the ability to access the functions from different programming languages.

Since each standard aids different types of graphical user interaction, the CCIS may require use of the following key standards:

- a. Graphical Kernel System (GKS), ISO 7942. GKS is intended for simple interactive applications where portability is more important than functional extensibility and performance. It supports a 2D graphics interface and a metafile that captures information at the virtual device interface to exchange pictures among applications and across separate environments. ISO 8651 specifies GKS language bindings for Fortran, Pascal, and Ada. (The DoD has funded the design of a graphics package for the WWMCCS Information System (WIS) project that extends the functionality of the GKS-Ada implementation [Foley 1986].) Bindings for C, Lisp, and Mumps are under development. It is also a FIPS standard, FIPS 120, for which NIST operates a Test Service. GKS-3D, ISO 8805, extends GKS to support 3D pictures, and language bindings for Fortran, Pascal, Ada and C are being developed under DIS 8806.
- b. Programmer's Hierarchical Interactive Graphics (PHIGS) Interface, ISO 9592. Intended for highly interactive applications, PHIGS provides multilevel graphics data structuring, geometry manipulation, 2D and 3D graphics, methods for real-time modeling, and a centralized hierarchical data store. ISO 9593 defines PHIGS language bindings for Fortran and Ada. Bindings for Pascal and C are under development. For use as a standard, FIPS 153, NIST is developing a PHIGS Test Suite, the first version of which is expected by the end of 1991. One extension, PHIGS+, provides for basic surface rendering. Another, PHIGS Extension to X (PEX), overcomes the limitation of X-Windows Version 11 to 2D. PEX takes the form of a protocol specification and any of PHIGS, PHIGS+, or GKS-3D libraries can be built on top of PEX. It supports CIE color standards.

4.7.2.2 Device Independent/Device Dependent Standardization

CCIS requirements explicitly state the need for use of "mouse" input devices, graphics terminals, large display devices, hard copy printers, and photographic-quality transparency copiers, and voice input/output devices. Independence of an application from a particular I/O device can be achieved by requiring the application to interface with a graphics package that produces device-independent output. This output is processed by a separate software module to generate the device-dependent commands needed for control.

Computer Graphics Interfacing (CGI) is an evolving standard, currently Draft ISO 9636, that defines a system-level interface to a virtual graphics device to provide a standard specification of control and data exchange between device-independent graphics software and one or more device drivers. CGI is the first graphics standard to deal explicitly with raster displays. It is expected to be published late 1991.

4.7.2.3 Color Standard

The manner in which output devices produce color is dependent on their hardware characteristics. Even devices of the same type may use different mechanisms. Consequently, standards pertaining to the specification of color are required to allow mapping of colors between different output devices. In order to fulfil the requirements for use of color the target profile will adopt CIE color standards. These are based on the premise that a color stimulus results from the proper combination of a light source, an object, and an observer. They are the 1976 CIE Uniform Chromaticity Scale (CIE-UCS) for color description and specification and the CIE L*, u*, v* (CIELUV) color space standard for colorimetry.

4.7.2.4 Rendering Standardization

Rendering is the process of translating geometrically described objects into displayable pixel-based pictures. In the absence of standards supporting device-independent rendering, the proprietary RenderMan interface offers the advantage of widespread use. Developed by Pixar, RenderMan is a scene description interface that partitions the generation of images into the distinct areas of interactive modeling and noninteractive rendering. It provides a basic set of 3D graphics functions, a hierarchical transformation stack with a full set of transformation operations, orthographic and perspective viewing transformations, and device independent image-size control. The RenderMan Shading Language provides user-extensible control over shading using geometric information, instead of the usual mathematical equation based on a simple model of the reflection of light. The RenderMan Interface Byte stream protocol provides both binary and ASCII encoding archive formats for network database transmission as well as file storage.

4.7.3 Human Factors Standards

The major goals of human factors standardization are to improve usability and productivity for users working with a variety of computer systems and applications, while also increasing comfort and well-being in the workplace with respect to the physical and psychological stressors. To help meet these goals and related CCIS requirements, the target profile will conform to the latest version of relevant standards and handbooks, including the following:

- a. MIL-STD-1472, *Human Engineering Design Criteria for Military Systems, Equipment, and Facilities.*
- b. MIL-H-46855B, *Human Engineering Requirements for Military Systems, Equipment, and Facilities.*
- c. MIL-HDBK-761A, *Human Engineering Guidelines for Management Information Systems.*
- d. MIL-HDBK-763, *Human Engineering Procedures Guide.*
- e. ESD-TR-86-278, *Guidelines for Designing User Interface Software.*

CCIS developers should also follow work being undertaken by the POSIX group, IEEE 1003.2 (in its User Portability Extension), and IEEE P1201 (Window Interface for User and Application Portability).

5. DISCUSSION

This section presents observations made during the course of the architecture task, along with recommendations derived from those observations. The discussion here is ancillary in that it is not part of the architecture or target profile. Instead, it concerns the process that surrounds the design of an architecture, the selection of its component parts, and the steps that need to be undertaken to follow up on the results presented in preceding sections. A number of open issues are presented as well. A general comment concerning these issues is that they all will need to be addressed in coming years.

5.1 OPEN SYSTEMS AND THE GENERIC CCIS ARCHITECTURE

5.1.1 Observations

The requirements upon which this effort is based are primarily from WWMCCS-related documents. The authors incorporated knowledge of other, primarily tactical, command and control systems. Nevertheless, the resulting architecture is believed to be suitable for a CCIS at any echelon. Furthermore, challenged on several occasions to identify unique aspects of a military CCIS, characteristics that differentiate it from other distributed information systems, none of deep significance could be identified. Generally, differences were in degree—the amount of protection and mobility, the kinds of networks supported, for example. That is, it is felt that every particular technology belonging in a military CCIS has an analogous potential for use in the civilian information systems community. The area most often questioned in this regard is security, and it may prove to be the case that security is an exception to the above generalization. While it is recognized that civilian systems require security, some even having security requirements that resemble those of the military, it is not to be assumed that technical solutions to the civilian security needs will meet military needs. For example, as more and more data is communicated electronically, including EDI and PDES data exchanged between different companies, the security prob-

lems parallel those of a military system. It is too early to know how this security issue will be decided because the problem specifications are not clearly defined yet.

Developers of future government and commercial systems see what technology currently allows and how it is changing. They are moving in essentially the same direction in their design of information systems. This has substantial benefit. Other architectural efforts, such as those of the Department of Defense's Corporate Information Management (CIM) initiative, the Internal Revenue Service's Tax Systems Modernization (TSM) program, and the Open System Environment program of NIST, will be common driving forces. Security aside, there appears to be very wide consensus among information systems architects on the direction technology is taking. The resulting solutions all look very similar. This is, in fact, the basis for optimism regarding the rapid development and adoption of standards in the industry. As long as everyone's vision of the future (at least the next 10 to 20 years of it) is similar, substantial cost savings to users will be seen if non-proprietary standards can be adopted for many aspects of information systems. The user community knows this and is beginning to exercise its collective muscle to force vendors to comply.

It is not just the CCIS architecture that is untested with respect to civilian requirements; civilian standards are untested with respect to CCIS requirements. There has been, of course, fairly extensive DoD involvement in the standards community, but not all of this has been from a CCIS perspective. An aspect of standards that may have significant impact on CCIS acceptability is performance. With hardware technology advancing so rapidly, the general perception, particularly in the non-CCIS community, is that performance problems are transient, to vanish with the next generation of hardware. For the CCIS community, the hope is that the adoption of open systems standards will lead to the same advantages, but there may be specific areas where such solutions simply do not apply.

For some standards, there are differences between American and International versions. This could lead to problems in interoperability with allies if systems only support the American standard. This may well be only a short-term problem for most areas, since the civilian community has analogous international interoperability requirements. A potential source of difficulty, however, could be U. S. Government versions or profiles of standards. At present, there is almost no experience that identifies which options are most appropriate to a CCIS nor any guidance on how to select options. The current version of U. S. GOSIP, for example, is incompatible with the current version of U. K. GOSIP.

5.1.2 Recommendations

Recommendation. The architecture needs to be tested against a broad sample of requirements in order to ensure that it is appropriate to the complete set of CCIS applications.

There are several ways this could be done: IDA or another contractor could be asked to review additional system requirements for a set of systems where the members of the set are selected in such a way that they are representative of the community as a whole. Alternatively, representatives of CCIS programs could be asked to review this document for suitability to their systems.

Recommendation. The contention that the architecture is not different in any substantial way from what is required of non-C² information systems should be tested in order to ensure that the most appropriate set of standards and profiles are selected for CCIS use.

Two aspects need to be considered. First, if commercial directions are ignored, substantial cost savings and other benefits could be lost. Second, if the commercial lead is followed blindly, significant CCIS requirements could go unmet. Neither case is acceptable.

Recommendation. Adopted standards should, consistent with the findings of additional study recommended in the preceding recommendations, be commercial standards.

It is expected that the greatest economic benefit will derive from using the same standards used by the civilian commercial community. This economic benefit is expected to derive in part from much greater flexibility in opportunity to obtain upgraded capabilities at low cost by using COTS software. For the CCIS community this means that new technology can be added to existing systems faster and less expensively than having to contract for their addition to military-unique systems.

Recommendation. Wherever the CCIS community is not currently involved in critical standards development, it should become involved, and where it is involved, stay involved.

Until it is clear that civilian standards are adequate for CCIS needs, the CCIS community should be vocal about making its requirements known to standards bodies and with NIST. This will be particularly important in areas related to international interoperability. In fact, given the large number of standards required for a CCIS, staffing a 100% complete effort is not feasible. Therefore, key efforts such as the APP, GOSIP, and POSIX should be given priority. At the same time many defense organizations such as the Navy's NGCR (Next Generation Computing Resource), the JIAWG (Joint Integrated Avionics Working

Group) and the STARS program (Software Technology for Adaptable, Reliable Systems) are very active in standards. Liaison with such groups could provide useful synergy.

Recommendation. Establish a national CCIS test bed.

Some problems described thus far derive from lack of experience while others reflect a lack of detailed knowledge of the suitability of a specific standard or combination of standards for CCIS use. Those details should be explored in advance of making commitments to specific programs. A CCIS testbed could serve other purposes as well:

- a. Support representatives to standards bodies by investigating the implications of proposed standards. In particular, new or changes to existing standards or profiles may need to be investigated for compatibility with existing usage and other standards.
- b. Evaluate new technology for its applicability to CCIS applications, for example, the role of multimedia in the CCIS at various echelons.
- c. Support prototyping for proposed applications, for example, the user interface.
- d. Support demonstration and validation of the evolving generic architecture.
- e. Support demonstration and validation of specializations of the generic architecture to specific systems.
- f. Evaluate competitive standards to understand their appropriate roles.

Recommendation. Once the evaluations recommended previously have been completed and the architecture is sound, require software developers, including COTS vendors, to apply accepted standards and establish firm, restrictive criteria for exceptions. Procurements must specify standards and give preference to conforming COTS products wherever appropriate.

Neither the architecture nor any of the recommended follow-on studies will do any good if the results are not used. Selected standards should be required now, with additions results from the test bed are obtained.

5.2 THE ARCHITECTURE MANAGEMENT PROCESS

5.2.1 Observations

The architecture effort described in this document looks at a time frame 5 to 20 years into the future. Because of the long lead times in military procurements, such a view is use-

ful in understanding the likely evolution of the systems being procured. But long-term planning for CCISs requires an evolving generic base. The value of the work reported here will decay if not maintained. Similarly, target profiles established for specific systems need to be based on the standards and technology available at the time the system is developed and maintained to support proposed evolution of the system.

In addition to the mid and long-range architectural view, a near-term view is also needed to support on-going efforts. The NIST APP is an excellent architectural base for today's open systems. It is anticipated that the APP will evolve to keep pace with the adoption of standards so that it will remain a good specification for what should be procured at any given point in time for immediate use. However, as noted above, without the involvement of the CCIS community, there is no assurance that the APP is now or will be entirely suitable for CCIS purposes.

Part of the process of managing the evolution of the architecture is ensuring that implemented systems are capable of evolving to keep pace with changing requirements. The areas selected for inclusion within the architecture are those believed to be important in shielding applications from such evolutionary changes. Users also need the same protection, and the areas most likely to change will affect applications and users alike. One example of special importance is improvements in input and output technologies. It is notoriously difficult to adapt application programs to such changes because applications typically contain very large amounts of input and output code designed to the specific devices in use. Another part of managing the evolution of the architecture is ensuring that adequate attention is paid to transitioning legacy systems. Deciding which systems to transition is a business decision, but one of the inputs to that decision should be guidance on the technical approaches that are appropriate. Additional guidance is necessary once the decision is made.

5.2.2 Recommendations

Recommendation. Establish an ongoing process to upgrade the generic CCIS architecture, especially with experience gained by implementors and users, feeding their ideas and problems back into the design process. Regular upgrades to adopt new technology and new requirements are needed, probably at intervals of two or three years.

Recommendation. Make sure that the CCIS community is adequately represented in the working relationship between DoD and NIST in the on-going evolution of the APP.

Recommendation. Establish a device-independent interface to shield the user interface from the particulars of input and output devices. The primary components of the user interface will be a UIMS, a windowing management system, and a set of graphics subroutine libraries. For the near term, these should be selected to conform with the NIST APP with the expectation that this will evolve to the IEEE P1201 Application Program Interface standard.

Recommendation. CCIS applications should be limited to the greatest degree possible to using only the application layer services and protocols of GOSIP, with procurements indicating a compliance requirement. Maintenance of application software or components should also be based on GOSIP application layer protocols and services.

5.3 THE RELATIONSHIP BETWEEN THE ARCHITECTURE AND APPLICATIONS

5.3.1 Observations

Policy and doctrine evolve as well as technology. An impediment to change in existing systems is the fact that policy is frequently embedded directly in application code. To change policy in response to changes in doctrine requires software modifications. The generality of the future CCIS and its ability to adapt rapidly to changes in policy and doctrine can be enhanced by an ability to store policy and doctrine in a form that can be dynamically interpreted. There is no experience, however, on how to do this in a way that ensures the required flexibility and security while ensuring enforcement of approved policy. This is true of almost every system in use today except, perhaps, small expert systems. The means to improve the situation is general but of a research nature.

Neither the generic architecture described in this document nor any other that does not include applications and mission areas can completely ensure that any system will meet the objectives as described below. These paragraphs show how the architecture supports the objectives and what additional is needed to accomplish them. In all cases, the developers of specific systems and applications will need to contribute to the successful attainment of the goals.

Interoperability is a combination of two characteristics, connectivity and understandability. The architecture supports connectivity by establishing standards for communications systems that facilitate common protocol usage. It supports understandability with the use of a common data management structure that includes the ability to exchange data descriptions as well as data values. It also supports understandability with data exchange standards for common application functions. Actual interoperation will require that com-

munications subsystems are physically connected, directly or indirectly, and that common protocols are used. It requires that COTS software vendors support the data exchange standards in their products for export as well as for import. It also requires that a common understanding of data is designed into the applications running on the systems.

Survivability is supported by the architecture through, among other things, standards that support replication of hardware, software, networks, and data, and by standards that support portability of software and data. Again, the ability to take advantage of these capabilities depends on actually making the applications portable, ensuring that the hardware is adequate to run any applications that may need to be run on it, and ensuring that the networks can be rapidly reconfigured. Survivability is also supported by mobility, flexibility, and security, all described below.

Flexibility with respect to procurement, deployment, and evolvability is supported by the use of non-proprietary standards that create the potential for truly portable applications, provided that the selected standards are adequately supported by vendors, and applications are developed without the use of any proprietary extensions to the standards or platform dependencies.

Mobility is supported by the use of a communications system that separates application requirements (reflected in the upper layers) from the networking requirements of the lower layers. Taking advantage of this requires that application developers not directly use communications services from the lower layers and avoid dependence on specific communications equipment. In some cases, this will not be possible, for example, in applications that depend on very high speed networks. The inability to use these applications when only low bandwidth radio is available must be considered in defining the procedures that use such an application. Mobility is supported by a data management structure that accepts temporary network partitioning as a normal condition. Operation under such circumstances is a clear requirement for some CCISs and will depend on the allocation of system resources.

Affordability will potentially be greatly enhanced by the use of standard interfaces and packages as well as by the opportunity to use COTS components. Note, however, that transition from proprietary technology to nonproprietary standards will be difficult. It can be facilitated with a good transition plan and guidance to application developers on how to make their programs more portable.

Security will be enhanced by the architecture but, as noted, it is still too early to evaluate the achievability and consequences of a truly pervasive multilevel secure system. Neither is it clear when such a level of operation will be feasible.

5.3.2 Recommendations

Recommendation. Support research on how to make policy and doctrine explicit within the system in order to provide flexibility.

This research should also address the management of doctrinal change.

Recommendation. Develop guidelines for application developers on how to make their applications portable.

This is an important step in actually achieving the desired results. This guidance should be considered a part of the architecture and evolve with its technical specifications.

Recommendation. Develop data definition standards for all CCIS mission areas and functions.

Much of this is currently underway. The recommendation is included to reinforce the significance of that work by indicating that it is also an important part of accomplishing the goals for this architecture.

5.4 THE TARGET PROFILE

The following sections provide additional commentary for the seven service areas.

5.4.1 Data Management

Experience with the use of data management standards in a distributed environment is lacking, but will be needed to provide guidance to implementors, for example, on how to avoid the pitfalls of vendor-unique extensions. Experience with the loosely coupled federated DBMS approach is also lacking. Successful application of this technology to the CCIS environment will take practice. Standards and implementations needed to support data management are still evolving as well. Still, experience with early versions should assist in the evolution and will serve as a foundation for experience with later versions.

5.4.2 User Interface

Well-established technology that should be exploited in the CCIS includes graphical user interfaces with windowing and direct manipulation capabilities. However, a common windowing system will not ensure a common user interface. Most of what needs standardizing is domain specific, e.g., terminology, symbology, and the use of color and graphics.

Such standards are likely to already exist within individual mission domains, but increased interoperation of systems may require the rationalization of these domain standards.

5.4.3 Network Services

The use of GOSIP protocols is one of the areas where additional experience is badly needed, particularly with respect to performance. In addition, variations between U. S. adoption of OSI profiles and profiles adopted by other governments will undoubtedly make it necessary for some CCISs to include additional protocols not covered by GOSIP. Guidelines on the implementation and use of such protocols is needed, and if NIST does not provide them, DISA will have to.

5.4.4 Operating Systems

Adequate guidance on how to write truly portable application software for POSIX-compliant operating systems is likely to take some experimenting. The problem is not unique to the CCIS community, but the CCIS community should be involved in whatever efforts are undertaken, to ensure that the experience applies adequately to the CCIS environment.

5.4.5 Security

To realize the full potential of interoperable systems, both within the CCIS community and with systems outside that area, multilevel secure systems will be needed. It is difficult to predict how long it will take to attain that capability, but in the meantime, we are no worse off for having open systems. Nevertheless, security standards are evolving and many other standards activities are now reflecting awareness of the need to incorporate aspects of security. This has both positive and negative results. On the positive side, standards are addressing the issue of security, but on the negative side, they are not proceeding in a coordinated fashion and thus, there may be different solutions to a given problem that will require a new effort to resolve the different results. For example, at least seven different groups are working on access control lists with different approaches on semantic details. One problem is that expert advice is not readily available in the voluntary standards bodies dealing with security issues.

In the meantime, the sources and uses of information at the highest classification levels should be analyzed. There may be a cost-effective partitioned system implementation that would bring the risk index to level two. Such a risk level requires hosts rated at Class B2 and there is good likelihood of COTS availability for this classification. However, partitioning should be considered an interim solution and investigation of fully integrated multilevel secure approaches should continue. Moreover, achieving MLS network security classes above a Class B1 rating in a *heterogeneous* system environment may be precluded by the issues of assurance (e.g., covert channels and the effect of cascading as described in the Trusted Network Interpretation [NCSC 1987]).

5.4.6 Programming Services

With the exception of language standards, there are no standards yet for software development tools that are supported by any appreciable consensus. This is not an accident. In general, there is very little agreement within the software engineering community on the best methods. There are numerous competing textbook methodologies for requirements development, design, testing, reviews, configuration control, project management, and software reuse, but in none of these areas has a clear winner emerged. As a result, it is not likely that any standards for software development tools will emerge in the near future. Standards development efforts are underway within IEEE and EIA, however, for standards for data interchange between tool sets. Current efforts to standardize the platform on which the tools execute (usually called the framework) are emerging. Proponents of an entity-relationship base for the framework have the lead at the moment with efforts to elevate PCTE to a standard, but other experts are equally adamant that the proper base is the object-oriented paradigm.

5.4.7 Data Exchange

There are many potential areas for data exchange standards to facilitate interoperability, but experience in using these standards within the CCIS context is lacking. For example, support for video teleconferencing and multimedia applications are stated CCIS requirements without reference to how these functions will be used, or even exactly what the requirements are in terms of amount of use, location availability, and other parameters

which can affect the ability of technology to meet requirements. At the moment, there are competing standards, with little guidance on which to use in specific situations.

5.5 DISCUSSION OF ISSUES

WWMCCS achieves effective communication through the use of a homogeneous system. However effective it is now, users will continue to demand new functions and features. WWMCCS network protocols and services have not kept pace with technology and are not flexible, adaptable, or extensible. Open systems technology holds the promise of a solution. However, its success will depend on the ability of old and new parts to interoperate.

The architecture is expected to evolve, and systems built to the architecture in the near term will need to evolve to keep pace. The key to graceful evolution is in anticipating change and designing the system in so that those changes are relatively easy to make. The decisions reflected in the architecture attempt to maximize the ability to adapt to changing technology, mission, and policy. It remains to be seen if the choices achieve that objective. One can hypothesize that technical breakthroughs or world events will disrupt orderly evolution. It is necessary to design the mission area applications with evolution in mind, or application maintenance and transition costs will completely overshadow the cost of the hardware and software components addressed by the architecture.

It would be preferable for all components of a system that are not mission unique were covered by international standards with support from vendors. The nature of requirements and the advance of technology are not likely ever to permit that completely. There will be times when no standards are available or when proprietary standards are the only alternative. It may very well be that virtually every CCIS requires extensions to parts of the architecture and waivers from other parts. Building usable systems that meet the goals of the architecture *to the maximum degree possible* will always be a challenge.

Just because a standard exists does not necessarily mean it is a correct solution to a technological problem. It may be appropriate to delay the implementation of new standards until they are proven in the marketplace. It is beyond the scope of the present effort to list these, but the developers of the 1995-97 CCIS will need to perform this study. The Department of Defense's CIM effort is encouraging rapid development of information processing standards for the Services as well as for the civilian agencies through NIST. There is a high probability that these standards will be incorporated by vendors into COTS products.

5.5.1 Network Services

Many COTS products support networking and communications. They include both hardware and software: cables, interface components, network cards, software for communications, software that implements protocols, and network management systems. The functions, capabilities, and complexity vary. OSI standards define protocols and services that can be used to establish communications capabilities within a heterogeneous distributed system. All communications-oriented products claim adherence to standards. As any systems integrator can testify, so many standards lack conformance tests, that "seamless" interconnection of the systems of different vendors is the exception rather than the rule. In the absence of trusted conformance certification, profiles or suites of compatible standards such as GOSIP will help smooth the way. GOSIP, however, may not fulfill the requirements of all systems. For example, it may never incorporate LAN protocols suitable for use on low bandwidth tactical radios.

5.5.2 Operating Systems Services

The IEEE Working Group on POSIX is the focal point of operating systems interface standards. It has broad industry and government involvement, but it remains to be seen if the resulting product is a standard that provides true portability of applications. The proliferation of subgroups has become a management nightmare with serious concerns that they may be working at cross purposes. Fault tolerance, a CCIS requirement, has not yet been addressed.

5.5.3 Programming Services

Existing CCISs were rarely programmed by their users. However, users are becoming increasingly computer literate and many have computer science degrees. This is creating a strong demand for programming capabilities within the CCIS. In addition, the system used to develop a future CCIS is likely to interoperate with the fielded system. These conditions will have significant security as well as technical implications. There is currently no standard tool interface, though the PCTE is rapidly emerging as the expected standard. It is not yet clear whether the PCTE intercommunications specification will conform to GOSIP or an equivalent set of standards.

There remain application areas for which the Ada language is not completely suited. These are addressed in a developing upgrade to the Ada standard, commonly referred to as Ada 9X, that is scheduled to be available in 1993. A significant number of older, non-Ada applications that will continue to be used for a time indicates that older programming languages such as Cobol, Fortran, CMS2, C, and Jovial will still need support.

Reuse of software is an immature technology. Although there are promising research and development results, there are still problems in implementation as well as some legal and technical issues remaining. In particular, reuse researchers and working groups are still addressing topics such as the following:

- a. Classifying a reuse library
- b. Performing quality control in library contents
- c. Describing modules that are more than simple software functions
- d. Integrating reuse in a development environment,
- e. Handling configuration and change management
- f. Customizing modules
- g. Handling the legal and procurement issues of licensing, data rights, and use royalties

5.5.4 Data Management Services

Major driving forces in the development of new data management technology include a desire to integrate geographically distributed databases in ways that enhance their utility and to provide (under a single data manager) data of a wide variety, including relational, object-oriented, audio/video, and knowledge bases. Such goals will lead commercial vendors to adopt standards and this will in turn eventually lead to open solutions that meet the requirements of a future CCIS. Mechanisms for accomplishing the integration of geographically distributed databases differ, based on the degree of autonomy required at the different locations. In general, a worldwide system is likely to include several approaches, from very tightly to very loosely coupled. The more tightly coupled systems are, the sooner the technology and the standards are likely to be available. Tightly coupled distributed systems are common today, and the use of open standards within them is increasing. The loosely coupled case is more difficult and open system issues are still under investigation.

The use of standards to define data management services will not guarantee interoperability. Barriers include extensions, profiles, undefined behavior, etc. Integrating data that is distributed requires knowledge of the structure and location of the databases. Within the CCIS community, organizational problems may be more difficult to solve than technical ones. Integration of geographically distributed databases is culturally different from the way data is shared today.

It appears to be possible to integrate different technologies with COTS models using modern technology (such as object-oriented and knowledge-based systems approaches), but it is likely to be necessary to provide autonomy at some (if not at all) node locations. Tightly coupled services are available today with open standards, but data integrity, confidentiality, and availability are of particular concern when integrating distributed database systems. Many of problems in providing such services remain open research issues.

5.5.5 Security Services

Current CCISs operate in system-high mode, but most data is not at the highest level of classification. To have information that is protected to a level above its security classification is operationally and economically undesirable. COTS products are beginning to reflect the demand for confidentiality. However, they are limited and generally emphasize discretionary access, with little consideration of mandatory access control. Integrity is addressed to some extent, and, because it is of concern to the commercial sector, it may receive more attention in COTS products. Availability remains a research issue. A widespread perception continues that there are substantial differences in the security concerns between the DoD and commercial environments. DoD participation in standards bodies addressing security is strongly recommended.

The following problems have been identified in evolving security standards:

- a. Many information systems standards today have a security component, but these are in an early stage of development, with few working proofs-of-concept. As a result, the security component may prove ill-specified or premature, and its use may adversely affect performance.
- b. Military security additions to information systems standards have appeared later in the field, or are still unspecified. Different groups may evolve incompatible standards that must interface. Seven different groups are working on access control lists (ACLs) with different approaches.

- c. Heterogeneous systems effects on security are mainly ignored. Specification of multilevel distributed systems security is incomplete and the solutions do not seem to be converging.
- d. Covert channels may preclude MLS above a B1 rating. However it is not unrealistic that B2 Operating Systems will be available in the time frame of WAM. Even some B2 database management systems may be available.

B2 requirements for covert channel analysis are not too stringent. At B2 level, DOD 5200.28-STD [30] (the Orange Book) requires "thorough search for covert storage channels" Techniques for identifying storage channels exist. Once they are identified, the Orange Book requires "determination . . . of the maximum bandwidth of each identified channel." The Orange Book also requires that the trusted computing base (TCB) be able to audit identified events that may be used in exploitation of cover storage channels. The greater difficulty for B2 may come from the system architecture requirements of the Orange Book: "The TCB shall be internally structured into well-defined largely independent modules." Without worked examples it is not clear what this statement requires. The operational meaning of this requirement should become clear once B2 systems have been successfully evaluated. At B3 the covert channel analysis requirements are expanded to include covert timing channels, a less well understood area. Consequently, B3 requirements are considerably more stringent than B2. At A1 the Orange Book requires that formal methods be used for analysis. In other words there is considerable difference between B2 and B3/A1. Developers should plan on using some B2 products, if only to facilitate later transition to MLS and to encourage vendors to build these products with open architectures in mind.

5.5.6 User Interface Services

User interface technology is changing rapidly. New forms of human-computer interaction and applications appear each year. The advent of multimedia representations has led to desktop video. Advances in flat panel and projection display devices are particularly important for a CCIS, where mobility and constrained operating environments have limited the use of CRT displays. Several prototype 3D virtual image displays have already been developed, but it is still immature and not recommended for use in the 1995 time period. CCIS

developers should keep abreast of this technology to ensure that it can be exploited when appropriate.

Prototyping should play a pivotal role in user interface design. Starting with the validation of the underlying user model and the look-and-feel standard, successive prototypes can support design trade-offs and expected user performance. Continuing evaluation of the user interface throughout the life of the system will make it possible to assess the impact of a changing user population in a real world environment. The majority of user interface work has focused on single-user systems but people work together in groups. The basic issues involved are *human-human interaction*. A new subdiscipline, computer support for cooperative work (CSCW), needs to be tracked by the CCIS community.

5.5.7 Data Exchange Standards

The push toward worldwide connectivity among almost all computers is creating substantial pressure for standards covering the data exchange area. Standards addressing the areas of electronic data interchange, unformatted documents, maps and geographic information, audio and video are rapidly approaching commercial acceptance, although not all problems will be solved by 1995. In some cases, overlapping standards compete for market dominance. It is too early to tell if one will prevail or, worse, if interoperability and openness will require supporting more than one standard in a given area. In other cases, while open standards are under development by international standards bodies, vendor support for these standards may be subsumed by de facto or proprietary standards already established.

If the Defense Mapping Agency continues to pursue a vector product standard different from the one under development by the rest of the federal government and gaining endorsements from the commercial community, problems may be created with respect to the use of COTS software for storing, retrieving, manipulating, and displaying maps and geographic information.

5.6 FINAL NOTE

The course of technology is rapidly changing and standards are seldom able to keep up. Despite the detailed guidance given above and in the appendices that follow, this document is to be taken as input and guidance for the designer of a specific CCIS. The designer will

have to discover the current state of standards, reevaluate and update needs, look deeply into the interoperation with current systems (both the *need* and the *technical aspects* of the interoperation), and finally, involve the anticipated user of the CCIS directly in the design process. Without all of these, risks cannot be bounded. With all, the likelihood for a successful, usable CCIS will be greatly enhanced.

APPENDIX A — DATA EXCHANGE STANDARDS

A.1 INTRODUCTION

Data exchange standards are needed for data that flows into, out of, or between all nodes of all Command And Control Information Systems (CCIS). Some such data products are exchanged with other systems outside the CCIS, some are used only within the CCIS community, and some apply equally to both situations. Three approaches were taken in developing this appendix: looking at the various types of data that are used in the command and control process, particularly data that must be shared with others, understanding the current status and direction of related data exchange technology and its application to CCIS, and identifying any standards that relate to the types of data that must be exchanged with other systems.

This document addresses the interchange among computing systems of seven types of data, formatted documents, unformatted documents, graphics, maps and geographic information, meteorological data, video and audio. Described within this appendix are the CCIS requirements for these data, related technologies, and standards that affect their interchange. Data compression, a subject that has direct application to several of the data types, is addressed in Section A.9

The primary focus in examining each of these data types is on the data formats and protocols used in exchanging the data between applications running on computing systems (see Figure A-1). While it is recognized that other hardware and software components of these systems contribute to the ability to make such exchanges, these are not the topics of this appendix. Thus, details of the communications and data management subsystems are not discussed except that some of the information contained in this appendix may be useful or even critical to the designers of those subsystems. Similarly, there may be implications in the data exchange technology for the input and output of such data that will influence the

design of the User Interface Management System (UIMS); these implications are identified, but user interface technology is not addressed here

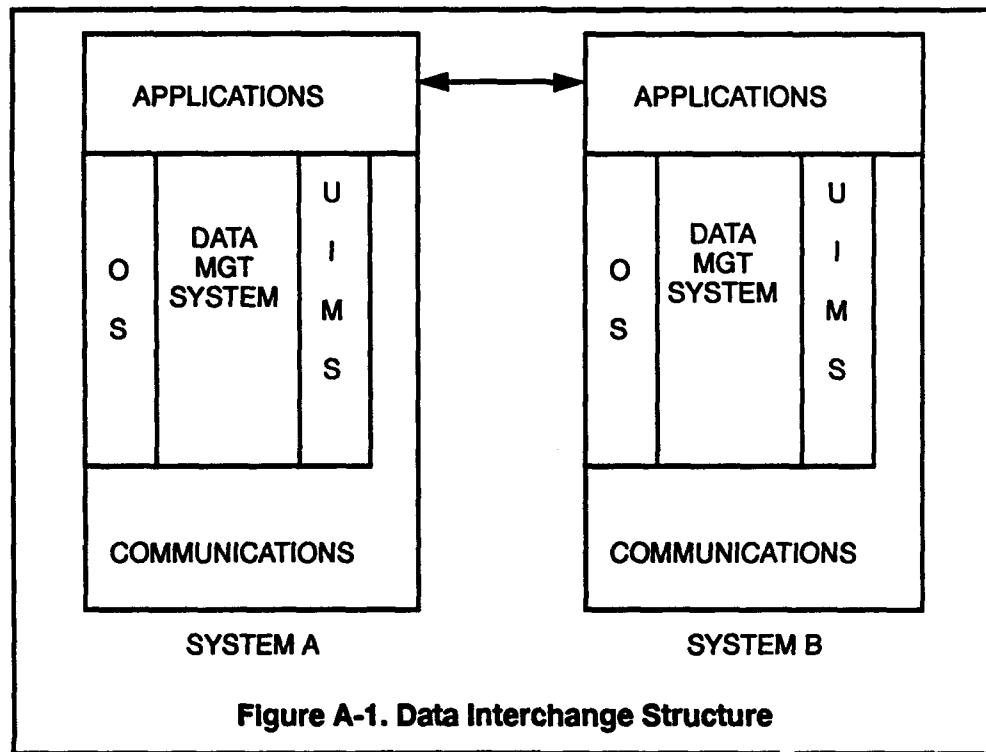


Figure A-1. Data Interchange Structure

It is important to keep in mind that the structure presented in Figure A-1 is relevant no matter how the system architecture is designed. The computing system itself could be a workstation, a minicomputer, a mainframe computer or a distributed system. For example, an application could execute on a computer different from the one hosting the data management subsystem. Some CCISs need to interchange data with systems that are not themselves CCISs. Thus, only one of the systems shown in Figure A-1 must be a CCIS. The other one may be a CCIS, another government system, or the system of a commercial organization. It is assumed that, sooner or later, where the timing will be determined more by policy than technology, virtually all of the computer networks in the world will be directly or indirectly interconnected. Therefore, regardless of where one draws the boundary between what is a CCIS and what is not, the parts of the CCIS along the edges, and indirectly all the other parts, will be connected to systems that are outside the boundary.

Specifically not included in this appendix, although it is recognized as a very important topic for CCIS interoperability, are standards for military-unique data, the bulk of the data exchanged in today's CCIS. Data such as unit status indicators, target tracks, and the many

other things that make up the domain of interest to the commander and his staff need to be, and are, the subject of a DoD-wide data element standardization process. The reasons for excluding discussion of these data elements here is, for the purposes of this architecture, it is sufficient to ensure that all necessary kinds of data can be exchanged, without concern for the specifics of what data is exchanged. The data elements in question belong to the application domain, which is generally being excluded from this study.

The remainder of this appendix is organized into the following sections:

- a. Data Exchange for Formatted Documents
- b. Data Exchange for Unformatted Documents
- c. Data Exchange for Graphical Data
- d. Maps and Geographic Information
- e. Meteorological Data
- f. Video
- g. Audio
- h. Data Compression
- i. Issues

A.1.1 Definitions

The types of documents as used in this appendix are defined in this section.

The terms *formatted* and *unformatted* documents are often misunderstood. Webster defines formatted as “to produce in a specified form or style” [Webster 1973]. Therefore, a formatted document would have a very rigid format that limits the type and amount of data that may be provided by the user. The formatted document usually consists of fields and field separators. Examples would be a purchase order form or a situation report where the user fills in certain fields. There may also be fields that allow some type of “free-form” text, i.e., description of item.

An unformatted document would allow the user some flexibility in format, data, and amount of data that may be included in a document. Although the structure may be pre-defined (i.e., must include titles, subtitles, paragraphs, etc.), the actual format is left to the discretion of the user. An example would be a scientific report or a manual.

There are some areas of difficulty in determining if a document is formatted or unformatted. Some documents may, in fact, be both formatted and unformatted. This would certainly be the case with a requisition form. Part of the document would be considered formatted—specific information requested for a field (i.e., name, order item). Another part of the document may be unformatted, for example, a justification for the requisition.

Several types of data are discussed in this document and defined as follows:

- a. **Graphics:** Any and all products of the cartographic and photogrammetric art. For this discussion, graphics will include map graphics (overlays), business graphics, raster graphics, schematic diagrams, etc.
- b. **Maps and geographic data:** All kinds of data describing the surface of the earth, including natural features (elevation, water depth, rivers, etc.) man-made features (roads, bridges, buildings, etc.) political boundaries (to all levels of detail) and terrain conditions (soil conditions, tree heights, etc.).
- c. **Meteorological data:** All weather-related data including observations made by people or sensors in the form of numbers, photographs, video images, etc., and forecasts.
- d. **Video:** Raster images taken from photographs or imaging equipment such as television cameras, and including both single frames and full motion.
- e. **Audio:** Digitized sound, primarily spoken or computer-generated words.

A.1.2 Related Standards

Table A-1 is a listing of all of the standards presented in this appendix. The listing is sorted by standard number and the section number, referring to pertinent discussion, is given.

A.2 FORMATTED DOCUMENTS

This section discusses the data exchange of formatted documents. The discussion includes a description of CCIS requirements, related technology, and associated standards. Examples of these types of documents include all existing formatted messages as well as

Table A-1. Related Standards

Standard Number	Standard Title	Section
ANSI p x 56	Digital processing of video signals—Video coder/decoder for audiovisual services at 56 to 1,536 kbit/s (p x 56)	9.3.1
ANSI X3.122-1986, ISO 8632	Computer Graphics - Metafile for the Storage and Transfer of Picture Description Information	4.3.2
CCITT H.261	Video Codec for Audiovisual Services at p x 64 kbit/s (p x 64)	9.3.1
CORINE	CORINE Data Transfer Specifications	5.3.2
DIGEST	Digital Geographic Information Exchange Standard (DIGEST)	5.3.2
DoD-STD-2167A	Defense System Software Development	3.3.1
DoD-STD-7935	Automated Data Systems (ADS) Documentation	3.3.1
FIPS Pub 70-1 (1986)	Specification for Representation of Geographic point Location for Information Interchange	5.3.1
FIPS Pub 103 (1983)	Codes for Identification of Hydrologic Units in the US and the Caribbean Areas	5.3.1
IEEE Project 949	Standard for Media Independent Information Transfer	2.3.2
ISO 7942	Information Processing Systems - Computer Graphics - GKS Functional Description	4.3.1
ISO 8613	Information Processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format	3.3.1
ISO-8879-1986(E)	Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)	3.3.1
ISO 9069	Information Processing - SGML Support Facilities - SGML Document Interchange Format (SDIF)	3.3.1
ISO 9735	Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)	2.3.1
ISO 10303	Product Data Interchange - Standard for the Exchange of Product Model Data (STEP)	4.3.2
ISO Draft Standard 9070	SGML Support Facilities - Registration Procedures for Public Text Owner Identifiers	3.3.2
ISO Draft Technical Report 10037	SGML and Text-Entry Systems - Guidelines for SGML Syntax-Directed Editing Systems	3.3.2
ISO Technical Report 9573	SGML Support Facilities - Techniques for Using SGML	3.3.2
SO/IEC 9592	Information Processing Systems - Computer Graphics Programmer's Hierarchical Interactive Graphics System (PHIGS)	4.3.1

Table A-1 (Continued)

Standard Number	Standard Title	Section
ISO/IEC JTC 1 DP 9636/1	Information Retrieval, Transfer, and Management for OSI	4.3.2
ISO/IEC JTC 15/SC 21 N 388	UN/EDIFACT Information Pack	2.3.2
ISO/IEC TR-9573	Information Processing - SGML Support Facilities - Techniques for Using SGML	3.3.1
ISO 10179	Document Style Segmentation and Specification Language (DSSSL)	3.3.1
ISO 10918	Joint Photographic Experts Group (JPEG)	9.3.2
ISO 11172	Moving Picture Experts Group (MPEG)	9.3.2
JBIG	Joint Bi-Level Imaging Group (JBIG)	9.3.2
MIL-D-28000	Digital Representation for Communication of Product Data: IGES Application Subsets	3.3.1
MIL-D-28003	Digital Representation for Communication of Illustration Data: CGM Application Profile	3.3.1
MIL-D-89000	Digital Terrain Elevation Data	5.3.1
MIL-D-89005	Digital Feature Analysis Data	5.3.1
MIL-A-89007	ARC Digitized Raster Graphics	5.3.1
MIL-M-28001	Markup Requirements and Generic Style Specification for Electronic Printed Output and Exchange of Text	3.3.1
MIL-M-28001A	Markup Requirements and Generic Style Specification for Electronic Printed Output and Exchange of Text	3.3.2
MIL-M-38784B	Manuals, Technical: General Style and Format Requirements	3.3.1
MIL-R-28002	Requirements for Raster Graphics Representation in Binary Format	3.3.1
MIL-STD-1840A	Automated Interchange of Technical Information	3.3.1
NATO ADat P-3, Part 1	NATO Message Formatting System (FORMATS): System Concept, Description And Management	2.3.1
NISTIR 88-4017	Standards for the Interchange of Large Format Tiled Raster	3.3.1
PB96-199759	Initial Graphics Exchange Specification (IGES)	4.3.1
SDTS	Spatial Data Transfer Standard	5.3.2
SPDL	Standard Page Description Language (SPDL)	3.3.1
VPS	Vector Product Standard	5.3.2

forms used in requisitioning, personnel management, and acquisitions from commercial suppliers.

A.2.1 CCIS Requirements

Two requirements affect the exchange of formatted documents. The existing use of formatted messages in CCISs. One objective of a generic CCIS is to eliminate formatted messages in favor of direct database exchanges of data. The replacement of formatted messages will be an evolutionary process requiring the oldest systems to use database approaches to maintain formatted messages during a transition period of many years [NIS ROC, 1983, 38]. Interoperability and closer coordination with systems not traditionally used directly by commanders and their staffs [JOPES ROC, 1983, 21]

Requirements for this exchange can be found in various Joint Chiefs of Staff and NATO Standardization Agreement (STANAG) documents. One example would be JCS Pub 1, *Department of Defense Dictionary of Military and Associated Terms*, which specifies exactly how changes to the Dictionary are to be submitted. There is a specific form that must be completed and returned either through regular mail or electronically. Another example would be the draft STANAG 4259 on Military Message Handling System. The functional requirements for this standard are based on the Allied Communications Publications 127 (ACP-127) with additional requirements drawn from procedures and security actions that were identified in the current NATO messaging network. The one requirement that was consistent throughout was compatibility among systems so that such items as forms would be formatted appropriately before and after transmission.

A.2.2 Related Technology

A.2.2.1 State of the Practice

CCISs today rely heavily on formatted documents for the exchange of data within and between themselves. Large volumes of standards (e.g., STANAGs and JCS Pubs) have been developed to specify the format, syntax, and content of these documents for general use. In addition, specific standards have been added to handle interoperability/compatibil-

ity among systems of different origin, i.e., within the Services or within international communities such as NATO.

A.2.2.2 State of the Art

This section discusses new technologies that appear to be ready for wide-spread commercialization. These technologies have been tested and proven to be successful. An assessment of the architectural significance of the technology is provided. Where possible, published industry projections on the penetration of the technology in the marketplace are provided.

The commercial and military information systems have in common the processing of various types of transactions in which the object of the transaction is to distribute data. The transactional nature of CCISs makes possible the use of commercial mechanisms for information transfer. Some of these commercial mechanisms are discussed below.

The Article Number Association (ANA) in the United Kingdom has established a Working Party on Trading Data Communications (TRADACOMS). This party was set up to enhance communication between the retail companies by designing and disseminating cost effective standards. The standards include agreed upon formats for such items as order forms and delivery notifications. In conjunction with these standards, ANA has published a *Manual of Standards for Electronic Exchange* which provides syntax rules and standards for coding various types of data. A similar effort, Electronic Data Interchange (EDI), has been developed by American National Standards Institute (ANSI) and International Organization for Standardization(ISO).

EDI is an example of a new technology that is spreading rapidly in the commercial community. EDI is the exchange of routine business transactions in a computer-processable format, covering such traditional applications as inquiries, purchase orders, order status, invoices, and financial reports. Its success is mainly attributed to the efforts of the large manufacturers and retailers. As small suppliers are brought on line, they are finding the new technology to their benefit. While the basic thrust of EDI has been the elimination of paper documents used for procurement, the major reason for the extensive push it is receiving today is that EDI permits the total integration of an organization's database with other organizations' databases with substantial cost savings. This is why EDI is now becoming more and more popular with the government.

The Computer-aided Acquisition and Logistic Support (CALS) initiative is perhaps the largest and best known proponent of EDI. It requires full compliance with the ISO EDI (ISO 9735) standard for digital delivery of technical information and interoperability among Department of Defense (DoD) systems. Major application areas include automation of technical manuals, computer-assisted design and spares acquisition. CALS will also use the Initial Graphics Exchange Specification (IGES) for engineering drawings, the Standard Generalized Markup Language (SGML) for automated publishing, and Computer Graphics Metafile (CGM) for technical manual illustrations.

As the CALS initiative continues to spread throughout the DoD, the opportunity to integrate the supply side of logistics with the demand side may become an important factor in integrating the command and control community with the logistics community. This is likely to happen both at the local level (a particular command center buying from the local economy) and the national level (the replenishment of military-unique supplies). The technology to accomplish this is rapidly becoming available. Numerous software vendors provide EDI software and thousands of EDI users are being added every year. In fact, the Army has begun to combine logistics with CCIS. If the Army is successful, the other Services will follow the Army's lead.

A.2.2.3 Future Research

The need to support EDI is essentially a need to interoperate with other military (logistics) and civilian systems. Interoperability with military systems will be via integrated databases and commercial EDI standards, the same ones used to interoperate among commercial systems.

A.2.3 Standards

Three areas are addressed in this section: current standards, current standards activities, and future standards activities.

A.2.3.1 Current Standards

This section addresses two standards being used in military and commercial applications:

- a. ISO 9735, *Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)*, 1988.
- b. NATO Allied Data Publication adat P-3, Part 1, *NATO Message Formatting System (FORMETS): System Concept, Description and Management*, November 1984.

ISO 9735. EDIFACT is a standard for data. It provides a set of principles which facilitate the electronic exchange of business data between manufacturers, exporters, wholesalers, distributors, etc. EDIFACT speeds up the flow of information and business transactions through the use of modern network services to transfer data.

The standard has a Document Application Profile (DAP) which consists of working implementation agreements. These profiles are used to transfer structured documents between equipment designed for word or document processing.

NATO ADat P-3, Part 1. In order to improve interoperability within the NATO command and control systems, the NATO Common Information Exchange Language (CIEL) was set up to provide rules, structures and vocabulary to standardizing messages suitable for use for information exchange between different national and NATO authorities and systems. Unfortunately, CIEL was never implemented. However, STANAG 5500 implemented the NATO Message Formatting System (FORMETS) which provides the rules, procedures, and vocabulary to be used in the construction of character-oriented message text formats for manual and computer-assisted operational environments. FORMETS is to be used for all formatted character-oriented messages unless specifically excluded through multinational agreement. FORMETS is primarily concerned with that part of a message which contains the thought or idea which the originator wishes to communicate. The syntax section provides very detailed rules governing the structure and arrangement of the components within a message.

Part 1 of this NATO standard outlines the procedures by which it is managed. FORMETS must be used for all formatted character-oriented messages within the NATO CCIS (NCCIS) unless it is specifically excluded by multinational agreement. The standard provides the basic structure (architecture) of FORMETS, its semantics, and its syntax. A formal description of the standard is provided in the standard's annex.

A.2.3.2 Current Standards Activities

This section identifies three standards activities currently under development.

- a. ISO/IEC JTC 1/SC 21 N 3885, UN/EDIFACT Information Pack, 11 October 1989.
- b. IEEE Project 949, Standard for Media Independent Information Transfer.
- c. CCITT X.425, Draft Standard for Integration of EDI with X.400, 1990.

ISO/IEC JTC 1/SC 21 N 3885. Electronic Data Interchange For Administration, Commerce, and Transport (EDIFACT) Information Pack is part of the ISO standard's work on the electronic exchange of data between systems and users. The pack defines EDIFACT, provides a history of EDIFACT, and lists the international EDIFACT structure and organization. The most helpful item in this pack is a list of key players in the EDIFACT movement. Names, addresses, and phone numbers of contacts in Western and Eastern Europe and North America are available.

IEEE Project 949. The Standard for Media Independent Information Transfer is being developed by the IEEE Computer Society. The purpose is to establish a common format or set of formats that will permit more convenient exchange of recorded information, independent of the medium of exchange.

CCITT X.425. A draft standard is also being developed in Europe that will integrate electronic data interchange with X.400. This will allow business partners to use the same standardized electronic process to exchange mail messages and EDI documents. The standard would enable users to support inter-enterprise EDI exchanges on their existing corporate-wide electronic mail network. EDI users will have access to the message management capabilities the electronic mail standard offers as well as the capabilities of the CCITT X.500 Directory standard. This draft standard, X.435, is now being reviewed by a CCITT study group.

A.2.3.3 Future Standards Activities

With the increase in the CALS effort and the emphasis being put on EDI, it is expected that most future standards activities involving formatted documents will come from these two arenas.

A.3 UNFORMATTED DOCUMENTS

This section discusses the exchange of unformatted documents. The discussion includes a description of CCIS requirements, related technology, and associated standards. Examples of these types of documents would be manuals and special reports. Electronic mail is not relevant to this discussion since it is covered in Appendix C, Network Services.

A.3.1 CCIS Requirements

Three requirements affect the exchange of unformatted documents:

- a. The need to produce and maintain unformatted documents.
- b. The desire to produce them on computers, particularly the same computers used for command and control purposes [JOPES ROC, 1983, 90 & 105].
- c. The desire to make use of electronic communications to disseminate them, both for review and for final distribution [JOPES ROC, 1983, 50 & 105].

A.3.2 Related Technology

A.3.2.1 State of the Practice

Most of the major CCISs today do not support the production and distribution of unformatted documents, but many "personal CCISs" developed by individuals or small groups for specific purposes do. More word processing today is done on computers by the document originator rather than a secretary or typist. Military personnel now rely on personal computers on field assignments as well. Personal computers are being procured for on-site use, such as aboard ships and in command posts, so users do not have to provide their own hardware and software. Where this is being done, the user is connected to networks permitting data exchange, including the exchange of unformatted documents.

A.3.2.2 State of the Art

The leading edge in unformatted documents is in the area of multimedia documents, including graphics, audio, photographs and full-motion video, along with television and

computer-generated simulations. The technology to support the generation of such documents is already available within a single system and promises to become more commonplace as the cost goes down and the ease of use goes up. More in-depth discussions of issues related to the distribution of the non-text components are included in Sections 4 through 9.

A.3.2.3 Future Standards Activities

Current products support the exchange of documents more through the ability to import and export foreign formats than through the use of common document exchange standards. Earlier attempts to create standards, such as the DoD Document Interchange Format (DIF) have not been very successful. The advent of multimedia documents has revived the interest in standards, but much work remains.

A.3.3 Standards

Three areas are addressed in this section: current standards, current standards activities, and future standards activities.

A.3.3.1 Current Standards

This section addresses those standards that are currently being used in military and commercial applications. The list is not meant to be inclusive, and represents those standards that we felt to have the most effect on data exchange within the DoD.

- a. DoD-STD-7935, *Automated Data Systems (ADS) Documentation*, 15 February 1983.
- b. MIL-M-38784B, *Manuals, Technical: General Style and Format Requirements*, 16 April 1983.
- c. ISO-8879-1986(E), *Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)*, 15 October 1986.
- d. MIL-STD-1840A, *Automated Interchange of Technical Information*, 22 December 1987.
- e. MIL-D-28000, *Digital Representation for Communication of Product Data: IGES Application Subsets*, 22 December 1987.

- f. MIL-M-28001, *Markup Requirements and Generic Style Specification for Electronic Printed Output and Exchange of Text*, 26 January 1988.
- g. MIL-R-28002, *Requirements for Raster Graphics Representation in Binary Format*, 20 December 1988.
- h. NISTIR 88-4017, *Standards for the Interchange of Large Format Tiled Raster*, 1988.
- i. MIL-D-28003, *Digital Representation for Communication of Illustration Data: CGM Application Profile*, 20 December 1988.
- j. DoD-STD-2167A, *Defense System Software Development*, 29 February 1988.
- k. IS 8613, *Information Processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format (ODIF)*, September 1, 1989.
- l. IS 9069, *Information Processing - SGML Support Facilities - SGML Document Interchange Format (SDIF)*, 1988.
- m. ISO/IEC TR-9573, *Information Processing - SGML Support Facilities - Techniques for Using SGML*, April 1988.

DoD-STD-7935. This standard provides guidelines for the development and revision of the documentation for Automated Data Systems (ADS) of applications computer programs and prescribes the standards and descriptions for each of the technical documents to be produced during the life cycle of an ADS. ADS is defined in the standard as "an assembly of procedures, processes, methods, routines, or techniques (including, but not limited to, computer programs) united by some form of regulated interaction to form an organized whole, specifically designed to make use of automatic data processing equipment" [DOD-STD-7935]. The objective of the standard is to provide managers of ADS projects with documentation of uniform format and content for review to assure the meeting of significant development milestones. It also provides ADS technicians with a standard record of technical information as a basis for coordination of later ADS development or use modification.

There are eleven technical documents described in the standard: Functional Description, System/Subsystem Specification, Data Base Specification, Computer Operational Manual, Test Plan, Implementation Procedures, Data Requirements Document, Program Specification, Users Manual, Program Maintenance Manual, and Test Analysis Report. A proposed outline and text format for each document type is provided in Section 3.0 of the standard.

MIL-M-38784B. This is a military specification approved by the DoD for use in developing technical manuals. Technical manuals are publications that contain instructions for the installation, operation, maintenance, training, and support of weapon systems, weapon system components and support equipment. Manuals prepared in accordance with this specification are intended for use in the operation and maintenance of equipment or for accomplishment of assigned missions. It covers the general style, structure, and format requirements for the preparation of manuscripts and reproducible copy for standard technical manuals and changes to those manuals. The only decision left to the author of a technical manual is the actual technical content of the manual; even the style of writing is specified (U.S. Government Printing Office Style Manual). The last sections of the specification discuss how to make changes to a technical manual, quality assurance provisions (readability, etc.), and preparation for delivery (packaging).

ISO 8879-1986(E). This standard was adopted but not mandated for use by the DoD on 4 January 1988. SGML standardizes the application of generic coding and generalized markup concepts. It provides a coherent and unambiguous syntax for describing whatever a user chooses to identify within a document. The language provides the following:

- a. An abstract syntax for descriptive markup of document elements.
- b. Concrete syntax that binds the abstract syntax to particular delimiter characters and quantities.
- c. Markup declarations that allow the user to define a specific vocabulary of generic identifiers and attributes for different document types.
- d. Provision for arbitrary data content.
- e. Entity references.
- f. Special delimiters for processing instructions to distinguish them from descriptive markup.

SGML was developed to solve such problems as device and system dependency, difficulties in integrating new technologies in the publications field, rekeying of data for multiple purposes, keying complexities, inability to develop multi-purpose training programs and the inability to exchange the structure of the text. The whole idea behind generic markup is to use nouns and adjectives to describe parts of a document to identify what something is, not how it looks. SGML is a language used to represent user-defined document structures rigorously so that a computer may process it.

SGML can be used to describe any type of textual data from a purchase requisition to a complicated mathematical equation. The Data Type Definition (DTD), however, narrows the meta-language for use with a specific application. A DTD is a concise statement of what elements, entities, and/or attributes are allowed in a particular document. Through the DTD, an application can rigorously define a class of documents such as a job guide or flight manual. Descriptive tags may be tailored to the application with no concern for the delivery method or output device.

Two companion standards are under development that some experts think will have a significant impact when combined with SGML. These are Document Style Segmentation and Specification Language (DSSSL) and Standard Page Description Language (SPDL).

MIL-STD-1840A. The purpose of this document is to standardize the digital interface between organizations or systems exchanging digital forms of technical information necessary for the logistic support of weapon systems throughout their life cycle. This standard addresses technical information and product definition data. The format, information structures, and transfer procedures are applicable in all cases where the information can be prepared and received in the form of American Standard Code for Information Exchange (ASCII) text files, product definition data files, raster image files, or graphics files. MIL-STD-1840A is the "parent" CALS standard that provides the rules for organizing files of digital data into a complete document.

Technical publications consist of text and associated illustrations. The files of a technical publication may consist of the following:

- a. A declaration file
- b. Text files (in ASCII) tagged to the contract
- c. Illustration files in IGES, CGM, or raster format
- d. Files in Page Description Language (PDL) form
- e. Other files (output specification file, special word file, etc.)

The standard dictates detailed requirements for the structure, content, and order of information. For example, the declaration file must precede the data files and provide information about the identifications, source, destination, classification, etc., of the document. The standard also specifies the file header records for the following:

- a. CGM data

- b. Document type definition
- c. PDL
- d. IGES data
- e. Gray scale
- f. Raster data
- g. Special word
- h. Output specification data

MIL-D-28000. This specification, adopted by CALS, is incorporated into a contract and identifies the requirements to be met when product definition data is delivered in the digital format of IGES as specified by ANSI standard, Y14.26M, *Digital Representation for Communication of Product Definition Data*. Product definition data is defined as “the totality of data elements required to completely define a product. Product definition data includes geometry, topology, relationship, tolerances, attributes, and features necessary to completely define a component part or an assembly of parts for the purpose of design, analysis, manufacture, test, and inspection” [ANSI Y14.25M]. The specification defines product data as “all data elements necessary to define the geometry, the function, and the behavior of a piece part or an assembly of parts over its entire life-span” [ANSI Y14.25M]. IGES is a specification that provides a neutral format for the representation and transfer of vector graphics data used for illustration purposes among Computer-aided Design (CAD) systems and application programs.

This specification defines the technical requirements for the exchange of digital product data in specific application subsets. These subsets are technical illustrations, engineering drawings, and electrical/electronic applications. The technical illustration subset addresses entities that support the exchange of figures and illustrations normally found in a technical publication. The emphasis is on visual clarity for human interpretation. The engineering drawings subset is used to encode product data being acquired in accordance with DoD-D-1000, *Engineering Drawings and Associate Lists*, for delivery in digital form. Exchange emphasis is on completeness, visual equivalency for human interpretation, and functionality of the received drawing model. The electrical/electronic applications subset addresses the representation and exchange of electrical and electronic products including printed wiring boards, printed wiring assemblies, hybrid microassemblies, cables, and wiring har-

nesses. Emphasis is on component and circuit element descriptions, their placement, their connectivity, and the routing of electrical paths.

MIL-M-28001. This specification establishes the requirements for the digital data form of technical publications. Data prepared in conformance to these requirements will facilitate the automated preparation, storage, retrieval, exchange, and processing of technical documents from heterogeneous data sources. The requirements set forth by this specification include the following:

- a. Procedures and symbology for markup of unformatted text in accordance with a specific application of SGML.
- b. SGML-compatible codes that will conform a technical publication to specific format requirements.
- c. Output control codes that will conform automated document processing functions to a uniform structure.

This specification establishes the requirements for the digital forms of all technical publications. Data files satisfying the requirements of this specification will be one of two types: Type I, MIL-M-38784A conforming technical manuals, and Type II, technical manuals conforming to other military specifications. Documents prepared in accordance with MIL-M-38784A and MIL-M-28001 must conform to the DTD defined in Appendix A of MIL-M-28001 and the output specification in Appendix C of 28001. The DTD and output specification for a MIL-M-38784A conforming manual do not have to be delivered with the tagged text. Technical manuals conforming to other military specifications may develop their own DTD but must use only those tags in the baseline tag set defined in Appendix B of MIL-M-28001. In this case, the DTD must be delivered with the publication along with a compatible output specification.

This specification addresses the five steps in the publication preparation process:

- a. Creating a DTD for publication control.
- b. Authoring the publication and inserting SGML markup tags.
- c. Verifying the syntax according to the rules of SGML.
- d. Using the output specification to compose the document so that the produced copy corresponds to the proper format and style.
- e. Generating a text presentation metafile in PDL to drive the display device.

The most important part of this specification is found in the appendices. The introduction to Appendix A provides an overview of the concepts behind the SGML standard, a brief tutorial on reading an SGML DTD, guidelines for using SGML tags, and DoD's SGML declaration. Appendix A specifies the role played by the DTD in an SGML implementation; a general description of DTD structure and content; the specific DTDs available for use in authoring, validating, and verifying an SGML-tagged technical document; and procedures for DTD development.

Two DTDs are also presented in Appendix A. The first DTD is for use when preparing a document that conforms with MIL-M-38784A. The second DTD uses the same elements as the first DTD with the addition of more subordinate paragraphs and steps. This DTD may be used for MIL-M-38784A non-conforming documents or as a model for the development of a more appropriate DTD. Both DTDs allow for four types of non-SGML data: IGES data, CGM data, CCITT Group 4 data, and system generated data.

Appendix B contains an alphabetical listing of all elements contained in the DTDs presented in Appendix A. Appendix C is a stand-alone document. It includes a document output specification (format and style guide) to be used for all applications of this specification. Although the format default values are set according to MIL-M-38784A, the values may be tailored to satisfy other format requirements. The appendix also provides an example of an SGML-coded source file and its output file.

MIL-R-28002. The DoD technical requirements for raster graphics are defined in this standard. This includes raster graphics that have been compressed to reduce file size and transmission time. These requirements have been adopted by CALS.

NISTIR 88-4017. This is a complement to MIL-R-28002. It defines proposed standards for the exchange of large formatted, raster documents. This standard has also been adopted by CALS.

MIL-D-28003. Computer Graphics Metafile (CGM) is widely available for authoring and graphics artwork stations. This directive defines the use of CGM for two-dimensional vector picture descriptions or illustrations in technical manuals and has been adopted for use in CALS.

DoD-STD-2167A. This standard provides the means for establishing, evaluating, and maintaining quality in software developed for weapon systems and its associated documentation. The contract agency is responsible for tailoring the software management process to

meet the needs of a particular project. The Data Item Descriptions (DIDs) associated with this standard describe a set of documents for recording information required by the management process. The standard encourages the production of deliverable data using automated techniques.

IS 8613. This international standard facilitates the exchange of documents. It addresses the logical structure and presentation of a document. It was originally designed for the exchange of office documents between different word processors. In the Office Document Architecture (ODA) world, document exchange is viewed from a communications perspective. ODA brings several information types together (i.e., document structure, layout, presentation, interaction and integration of text and graphics) under a single standard. This is different from SGML which deals only with text and then uses other standards to accomplish graphics, etc.

The standard describes a document in terms of its logical structure (document profile) or its layout structure (page layout) or both together. There are eight parts to the standard:

- a. Part 1 is the introduction and general principles.
- b. Part 2 defines the formal ODA. This formal specification is concerned with the descriptive representation of documents, a document processing model, the document structures, the appropriate reference models, and three document architectures classes. It also provides examples of document architecture levels, document structures, and document attributes.
- c. Part 3 is the document processing reference model.
- d. Part 4 defines the purpose of the document profile and specifies the attributes which constitute the document profile.
- e. Part 5 of the standard specifies a second method of representation and interchange using an Office Document Language (ODL) and the SGML Document Interchange Format. ODL is an application of the SGML and may be used to represent a document structure in accordance with ODA in SGML.
- f. Part 6 defines the character content architectures that can be used in conjunction with the document architecture defined in Part 3. It also defines the internal structure of content conforming to the character content architectures. A content layout process is described which, together with the document layout process,

determines the layout of character content in basic layout objects and the dimensions of these objects.

- g. Part 7 defines the raster graphics content architectures that can be used in conjunction with the document architecture defined in Part 2.
- h. Part 8 defines the geometric graphics content architectures and the presentation attributes applicable to this architecture.

This international standard has been adopted for use by the Federal Government for incorporation in the Application Portability Profile (APP), and by the DoD for inclusion in the CALS initiative.

IS 9069. This ISO standard is used solely for the exchange of SGML document. It provides minimal information on data exchange outside of SGML. It does establish a format for exchange such that data streams, etc., are set up alike along with the encoding rules for the exchange of SGML documents.

ISO/IEC TR-7935. This technical report establishes rules for the markup of documents for publication, exchange, and analysis of documents prior to writing the formal document type definition. This technical report, which is complementary to ISO 8879, provides a three-step analysis phase:

- a. Identify elements and attributes requirements.
- b. Identify hierarchical structure of elements.
- c. Evaluate the use of short references.

The report provides the user with several complete examples and detailed descriptions for all the tags.

A.3.3.2 Current Standards Activities

This section addresses two areas where new standards are currently under development: SGML and Profile Alignment Group for ODA (PAGODA) [Open Systems 1990]. Where known, an estimated completion date is included.

Three SGML standards are under development:

- a. ISO Technical Report 9573, *SGML Support Facilities Techniques for Using SGML.*

- b. ISO Draft Standard 9070, *SGML Support Facilities Registration Procedures for Public Text Owner Identifiers*.
- c. ISO Draft Technical Report 10037, *SGML and Text-Entry Systems - Guidelines for SGML Syntax-Directed Editing Systems*.
- d. MIL-M-28001A, *Markup Requirements and Generic Style Specification for Electronic Printed Output and Exchange of Text*, 1990.

A new version of MIL-M-28001, MIL-M-28001A, was released on July 20, 1990. It specifies the CALS requirements for the digital data form of page-oriented technical publications. Data prepared in conformance to these requirements will facilitate the automated storage, retrieval, interchange, and processing of technical documents from heterogeneous data sources. The requirements set forth by this specification include:

- a. Procedures and symbology for markup of unformatted text.
- b. SGML compatible codes that will support encoding to specific format requirements.
- c. Formatting Output Specification Instance (FOSI) based on the output specification.

The PAGODA was formed in 1988 to coordinate and align the work done by three regional workshops (National Institute for Standards and Technology (NIST), AOW, EWOS) and by CCITT in ODA profiling. The goal of PAGODA is to ensure that revisable documents can be exchanged electronically worldwide. Each regional workshop produced one of the three ODA Document Application profiles that will be submitted to ISO as International Standard Profiles (ISPs) for ODA. Three of these, now called Office Document Formats (ODF), have been completed by PAGODA but have not yet been accepted by ISO. Still to be done is work related to aspects of conformity and the behavior of implementations.

A.3.3.3 Future Standards Activities

Future activities will need to include the merging/coordination on standards in the SGML and ODA worlds. The most popular view is that SGML will become a part of the ODA standard. PAGODA may be the beginning of this effort.

A.4 DATA EXCHANGE FOR GRAPHICAL DATA

Graphics, as defined by JCS Pub 1, is any and all products of the cartographic and photogrammetric art. Graphical data interchange can be specified in terms of a file format that can be created independently of any device requirements and then translated into the formats needed by specific output devices, graphics systems, or computer systems. For this discussion, graphics will include map graphics (overlays), business graphics, raster graphics, schematic diagrams, etc. Digitized maps and video are discussed in detail in Sections 5 and 7, respectively.

A.4.1 CCIS Requirements

Existing military CCISs support the generation and display of graphics. This capability will continue to be required in future CCISs. However, graphics are not generally distributed. Instead, the data is distributed and the graphics are regenerated at each location where they are needed. The need to distribute graphics derives more from the need to handle more complex types of graphics, such as raster graphics (scanned images) and schematic diagrams, and unique sources of images, such as business graphics generated by an application that was developed by the user [NIS ROC, 1983, 13]. Distribution of these types of graphics today tends to be slow, unreliable, and often expensive.

A.4.2 Related Technology

A.4.2.1 State of the Practice

Graphics play a major role in military applications. Previously, the exchange of graphical data was limited to the exchange of flat files where the assumption was that the recipient would run the same graphical software as the originator. Today, there is a growing interest in using graphics for simulation purposes. The Army has developed an Army Human Engineering Laboratory (HEL) where aviation and air defense teams can carry-out missions without real gunplay. The Marine Corp has developed a Digital Communication Terminal (DCT) that transfers digital information while running certain simulation programs.

Graphical data is also exchanged through postscript files, scanners (i.e., bit maps) and computer-to-computer exchange of facsimile transmissions.

A.4.2.2 State of the Art

The Perseus project is now undergoing beta testing at the NIST. This is a multimedia data base that will let students browse ancient Greek literature, history, and archaeology. Its underlying technology, combination of next-generation computer simulation that simultaneously employs graphics, text, bit-mapped or raster images, is being investigated by the DoD for use in their simulation projects.

A.4.2.3 Future Research

The graphics used in the Perseus project may be beneficial to CCISs. However, there are some problems now with porting and electronic exchange of data from these systems. Future research needs to be done in this area to better utilize graphics in simulation programs and educational purposes.

A.4.3 Standards

Three areas are addressed in this section: current standards, current standards activities, and future standards activities.

A.4.3.1 Current Standards

There are several terms related to the graphical world that are pertinent to the exchange of graphical information:

- a. **Computer Graphics Reference Model** defines a basic architecture and terminology for computer graphics. Areas addressed in the model include environment, primitives, pictures, meta-files, and archives. Four types of environments are also present: application, virtual, logical, and physical. Specific standards related to this model are discussed later in this section.

- b. **Computer Graphics Metafile (CGM)** standards refer to the file format suitable for the storage and retrieval of picture information. The file format includes a set of elements which can be used to describe pictures so that they are compatible between systems of differing architectures and capabilities.
- c. **Graphics Kernel System (GKS)** specifies a language-independent nucleus of a graphics system. So that GKS can be integrated into a specific programming language, GKS is embedded in a language-dependent layer obeying the particular conventions of that language.
- d. **Programmer's Hierarchical Interactive Graphics System (PHIGS)** provides a set of functions for the definition, display and modification of two- and three-dimensional graphical data. It also provides a set of functions for the definition, display and manipulation of geometrically related objects.
- e. **Computer Graphics Interface (CGI)** is described later in this section in reference with specific standards.

The following list of standards addresses the storage, manipulation and transfer of graphical information:

- a. ISO 8632, *Information Processing Systems - Computer Graphics - Metafile for Storage and Transfer of Picture Description Information*, 1 August 1987.
- b. ISO 7942, *Information Processing Systems - Computer Graphics - GKS Functional Description*, 15 August 1985.
- c. ISO/IEC 9592, *Information Processing Systems - Computer Graphics - Programmer's Hierarchical Interactive Graphics System (PHIGS)*, 1 April 1989.
- d. PB86-199759, *Initial Graphics Exchange Specification (IGES)*, Version 3.0, April 1986 (Also known as ANSI Y14.26M - 1989 Version 4).
- e. ANSI X3.122-1986, *Computer Graphics - Metafile for the Storage and Transfer of Picture Description Information*, 27 August 1986. (same as ISO 8632 - all four parts).

ISO 8632. This standard was developed for producing CGM in order to:

- a. Allow picture information to be sorted in an organized way on a graphical software system.
- b. Facilitate transfer of picture information between different graphical software systems.

- c. Enable picture information to be transferred between graphical (hardware) devices.
- d. Enable picture information to be transferred between different computer graphics installations.

There are four sections to this standard. Section 1 of ISO 8632 provides a file format suitable for the storage and retrieval of picture information. It consists of a set of elements used to describe pictures in a way that is compatible between systems of different architectures and devices of different capabilities and design. It also defines the form (syntax) and functional behavior (semantics) of a set of elements that may occur. Lastly, the standard describes all of the elements found in CGM, which is a collection of elements from a standardized set. Part 2, character encoding of the CGM, provides a representation of the metafile syntax intended for situations in which it is important to minimize the size of the metafile or transmit the metafile through character-oriented communications services. The encoding uses compact representation of data that is optimized for storage or transfer between computer systems. Part 3 concentrates on binary encoding. It includes a representation of the metafile syntax that can be optimized for speed of generation and interpretation, while providing a standard means of interchange among systems. The standard uses binary data formats that are much more similar to the data representations used within computer systems than the data formats of other encodings. Part 4 of this standard is concerned with clear text coding for all elements in a Computer Graphics Metafile. It provides detailed descriptions and samples for encoding CGM.

ISO 7942. This standard consists of a set of functions for computer graphics so that computer-generated pictures can be produced. It fulfills the requirement for a language to program two-dimensional graphical objects that will be displayed or plotted on appropriate devices (raster graphics and vector graphics devices). The standard includes functions for storage on and retrieval from an external graphics file. It discusses various attributes of GKS and has numerous annexes that cover areas such as function lists, error lists, interfaces, metafile structure, sample programs, and functions summary.

ISO/IEC 9592. This international standard enables graphical data to be stored in a hierarchical data store and draws extensively on GKS for its model and functionality. Information in the data store can be inserted, modified and deleted with the provided functions. The standard allows application programs using dynamic hierarchical graphics to be easily portable between installations, aids the understanding and use of dynamic hierarchical graph-

ics methods by application programmers, reduces program costs and time, and serves manufacturers of graphics equipment as a guideline in providing useful combinations of graphics capabilities in a device.

There are three parts to the standard. Part 1 is the functional description where the specific functions for computer graphics programming are presented. Part 2 provides a file format suitable for the storage and retrieval of PHIGS structure and network definitions. The file format consists of a set of elements that represent structure elements in a way that is compatible between systems of different architectures and devices of differing capabilities and design. Part 3, clear-text encoding of archived files, provides a representation of the archive file syntax that is easy to type, edit and read. This representation allows an archive file to be edited with any standard text editor, using the internal character code of the host computer system.

PB86-199759. This specification was developed by the NIST to define a format for the creation of a file which enables data found in computer-assisted design and computer-assisted manufacturing (CAD/CAM) systems to be exchanged or archived. It also provides for increased capabilities in geometry and non-geometry data exchange. The specification includes a file structure format, a language format and the representation of geometric, topological, and non-geometric product definition data in these formats. The specification is very specific about both the ASCII and binary formats that must be defined to represent IGES data. There is also a detailed discussion of the geometry entity types available and the capabilities for representing non-geometry entity types.

This specification has also been defined as ANSI standard, ANSI Y14.26-1989, by the Association of Structural and Mechanical Engineers (ASME). It is also used in the DoD CALS initiative to specify transmission of technical documents and engineering drawings in a device independent manner.

ANSI X3.122-1986. This ANSI standard was adopted by ISO as ISO 8632. The only difference is that the ANSI standard is one document, where the ISO standard is printed as four separate parts.

A.4.3.2 Current Standards Activities

This section addresses what standards are currently under development with an estimate of the completion date.

- a. ISO/IEC JTC 1 DP 9636/1, Information Retrieval, Transfer, and Management for OSI.
- b. ISO 10303, *Product Data Interchange - Standard for the Exchange of Product Model Data (STEP)*, Draft Proposed Standard.

ISO/IEC JTC 1 DP 9636/1. This draft proposal under consideration by ISO is concerned with developing a computer graphics interface that is device independent. The functions for the control and data exchange of this interface are described in the proposal. The interface may be implemented as a software-to-software interface or as a software-to-hardware interface. There are several parts to the proposal, but Part 1 is where the graphics standards reference model is discussed. This model is strongly tied to GKS, (ISO 7942), 3-D (ISO/DP 8805), and PHIGS.

ISO 10303. Previously known as Product Data Exchange Specification (PDES), this standard defines a complete product life cycle including all aspects of describing technical diagrams and documents in a neutral format for transmission over communications networks and processing by numerically-controlled machine and assembly tools. Product Data Interchange - Standard for the Exchange of Product Model Data (STEP) may be used in total life cycle descriptions of engineered products that can be implemented on advanced manufacturing systems.

STEP is still in the draft stage and may undergo revision at any time. Many of the components of the specification have not been defined, although the projected date for the majority of the component specifications to be ready is early 1992.

A.4.3.3 Future Standards Activities

Three upward-compatible addenda to CGM are being considered. These additions include adding a global symbol capability, adding three-dimensional geometry extensions and adding improved engineering drawing capabilities, such as better control over fine details of line drawings [NIST 1990].

A.5 MAPS AND GEOGRAPHIC INFORMATION

This category includes standard military and civilian maps, and the contents of geographic information systems (GIS).

A.5.1 CCIS Requirements

A number of existing CCISs make use of digital maps and geographic information today, and such use is expected to grow. Explicit requirements for the processing and interchange of maps and geographical information are stated directly in the requirements for the following capabilities:

- a. To display and transfer a working color map between two or more headquarters [NIS ROC, 1983, 8]
- b. To change map features, post symbols and have a zoom capability [NIS ROC, 1983, 8]
- c. To receive, store, process, display, and integrate all environmental data [NIS ROC, 1983, 12]

In addition, the general requirements for simulations [JOPES ROC, 1983, 65 & 68; JCS PUB 6-03.10 Vocoder. I Annex K, III-8] and analyses [NIS ROC, 1983, 6] are likely to include the use of maps and geographical data for simulating air and ground planning alternatives or for analyzing terrain features for possible operations. For example, the US Army Corps of Engineers Engineering Topographic Laboratories (USAETL) provided the Army and Marine Corps with systems for decision support, terrain analysis, imagery analysis including map generation, precision location data, climatology, and terrain masking during Operation Desert Storm. They are also working on a number of artificial intelligence systems that would provide direct terrain reasoning support to mission planning in such areas as route planning and mine field site prediction. These systems are all based on GIS databases and technology.

A.5.2 Related Technology

A.5.2.1 State of the Practice

Digital cartographic and geographic information systems have existed for several years, and have found a wide variety of uses in civilian and military applications. The development of computing, storage, and display technology of sufficient speed, capacity, and resolution have made it possible to describe the earth and its features with great precision. The primary impediments to widespread use of such information has been (1) the data collec-

tion process, and (2) standards for information sharing. In some cases, data collection must start from scratch in areas where no local agency has had responsibility for maintaining geographic information. In other cases, the data exists but only on paper, and in all cases there is a desire to add data elements not presently in the database.

Most of the data collection has been carried out by local and national authorities who now want to expand their holdings by exchanging this information with each other. This has lead to the interest in standards. Numerous national and international bodies have already taken steps to develop standards for the representation of the earth and the things on it, and some commercial products have been built using these standards. For example, several companies sell mapping systems and digital maps that are used for such applications as dispatching emergency vehicles, sales analysis, utility routing and other business and government needs. Also, for several years there have been products targeted toward the military market for the display of digital maps and associated geographic information. These generally have a limited storage capacity, and are intended for use by a field unit in a limited area. Top-of-the-line automobiles are now available with digital map displays that are driven by CD/ROMs (compact disks/read-only memory).

A.5.2.2 State of the Art

The main areas of current research and development activities related to digital cartographic and geographic information are in the areas of database development. For example, a prototype of the Digital Chart of the World (DCW), developed by the Defense Mapping Agency (DMA), is currently under review. DCW, when complete, will contain a complete 1:1,000,000 scale map of the world that includes a large amount of topological and cartographic data, and will occupy thirty CD/ROMs. To put such a database together requires rationalizing data from a wide variety of sources. This, in turn, has depended on the development of algorithms for converting data in a number of different coordinate systems into a common form. Toward this end, a handbook on data transformations is being produced by DMA.

A.5.3 Standards

A.5.3.1 Current Standards

The standards listed below generally fall into two categories, encoding standards and exchange standards. The encoding standards relate to the subject matter of cartographic and geographic data. The exchange standards describe how the exchange media are to be organized. Typically, the exchange standards reference the encoding standards for the specification of values of the fields in records. At present, all the exchange standards are designed for removable media, such as magnetic tape. There are no standards establishing communications protocols for exchanging cartographic and geographic data. However, the file specifications of the exchange standards could be used with general purpose file transfer protocols, but no cases of such usage are known.

a. Military:

- (1) MIL-D-89000, *Digital Terrain Elevation Data*
- (2) MIL-D-89005, *Digital Feature Analysis Data*
- (3) MIL-A-89007, *ARC Digitized Raster Graphics*

b. Other Government:

- (1) FIPS Pub 70-1 (1986), *Specs for Representation of Geographic Point Location for Information Interchange*
- (2) FIPS Pub 103 (1983), *Codes for Identification of Hydrologic Units in the US and the Caribbean Areas*

c. Commercial: None.

A.5.3.2 Current Standards Activities

a. Military:

- (1) *Digital Geographic Information Exchange Standard (DIGEST)*. This standard is currently under development by the ten-nation Digital Geographical Information Working Group (DGIWG). They are expecting to submit it to NATO to become a STANAG. Some discussions have taken place within DGIWG regarding submitting DIGEST to ISO, but there is no plan for this

at the moment. Their present concern is magnetic tape exchanges, with electronic communications exchanges possible in the future. The position of the DGIWG is that DIGEST is intended for standardizing exchanges of data between map-producing agencies like DMA, and not between operational units; also that standards governing exchanges between field systems are the responsibility of the system development organization. This is a traditional view in military systems development organization, and leads to substantial interoperability problems, particularly intra-national. The official position notwithstanding, the DGIWG is encouraging the distribution of DIGEST by its member nations to the widest possible audience, including the services and civilian users.

- (2) *Vector Product Standard (VPS)*. Developed by the Defense Mapping Agency (DMA), VPS is currently in prototype stage, nearing final. Although the standard is being distributed to the civilian community, there are currently no plans to offer VPS as a civilian standard. VPS was originally developed as an alternative to SDTS (see below), but it is becoming apparent that SDTS is much more likely to garner industry acceptance and vendor support. It is likely that DMA is going to have to rethink its expectations to base all future products on VPS.

b. Other Government:

- (1) *Spatial Data Transfer Specification (SDTS)*. Issued by US National Committee for Digital Cartographic Standards, a multi-agency working group headed by the U.S. Geological Survey (USGS). DMA was an original participant in the development of this standard, but dropped out in favor of their own activities. SDTS is expected to become a Federal Information Processing Standard (FIPS) in mid -1992.

Standards under development by USGS include aquifer names and geologic unit codes, classification of wetlands and wildlife services, EPA parameter codes, codes for taxonomic identification of flora and fauna, land use and land cover codes, public land survey codes, and cartographic attribute/feature codes.

c. Foreign:

- (1) UK Ministry of Defense (status unknown): digital terrain elevation data, and digital feature analysis data.

- (2) CORINE (a project of the European Community): CORINE Data Transfer Specifications
- d. Commercial: None.

A.6 METEOROLOGICAL DATA

This category includes all forms of weather data, including both observations and forecasts.

A.6.1 CCIS Requirements

Weather forecasting plays an important part in command and control decision making. At present, meteorological data is exchanged over special purpose networks, including teletypes and dedicated satellites. The requirement to support meteorological data exchange on CCISs stems from general requirements for integration and interoperability, as well as likely future desires to incorporate meteorological information more directly in analyses taking place within the CCIS. Specifically, there is a stated requirement to be able to receive, store, process, display, and integrate all environmental data [NIS ROC, 1983, 12].

A.6.2 Related Technology

A.6.2.1 State of the Practice

There is a world-wide network permitting the national weather agencies to exchange observational data. This network, and the related networks within the U.S., are based on a teletype exchange mechanism and radio broadcast of satellite images.

A.6.2.2 State of the Art

Within the US, three federal agencies are involved in the collection, processing and distribution of meteorological data, the National Oceanic and Atmospheric Administration (NOAA), the Federal Aviation Administration (FAA), and DoD. These three organizations are currently cooperating in the modernization of their capabilities in this area. NOAA's

National Weather Service is leading the effort with the acquisition of new sensors, new computers, new algorithms, and a new communications network. Some of the new equipment is currently being installed for a demonstration phase prior to full scale implementation. The computational component of this work is called the Advanced Weather Interactive Processing System (AWIPS). The Air Force is also developing a system called the Advanced Meteorological Processing System (AMPS).

As part of the AWIPS program, NOAA is developing a network called NOAAPORT both for internal data distribution needs and for distribution of its products to outside organizations, including the military.

A.6.2.3 Future Research

The Air Force is currently conducting research on the following topics:

- a. How to organize data in more cohesive ways to speed distribution.
- b. Automated analysis.
- c. Two- and three-dimensional graphic displays of observational data for use by forecasters.

A.7 VIDEO

Today's CCISs already routinely use video displays for displaying text, business graphics, and map graphic overlaid on paper maps. This section addresses additional uses of video for display of photographs, television, and a wide range of computer-generated still and motion images from weather forecasts to flight simulations.

A.7.1 CCIS Requirements

CCIS planners are discussing a variety of applications that will depend on video technology [NIS ROC, 1983, 5; JCS PUB 6-03.10 VOL II ANNEX B, B-8]. The major use could be for multimedia information exchange within a commander's situation briefing. Full motion video is likely to be required in at least some training and intelligence gathering applications. Training video will likely be a combination of simulation-generated and pre-recorded television. Intelligence gathering today includes TV monitoring from remote

locations. In the future, the CCIS may be called upon to store and transmit such images to analysts at distributed locations. Meteorological data collection and weather forecasts may also take advantage of full motion video in the near future.

Still picture applications include multimedia documents, such as training and maintenance manuals, medical data, particularly x-ray and similar images, and intelligence.

A.7.2 Related Technology

A.7.2.1 State of the Practice

There is not much use of video imaging technology in CCISs today because the television-related technology is too new and the simulation technology is too expensive. In both areas, almost all current use is in dedicated systems. In the case of television-based imaging, this is rapidly changing as digital video technology is becoming available for personal computers and workstations. Similarly, as workstation computers become more powerful, they are being used more and more for simulations. Still, at present, these systems are largely dedicated to a single application. Whatever data exchange is taking place is largely in the form of exchanging the generation algorithms and flat data files, rather than employing any specific data exchange protocols.

A.7.2.2 State of the Art

An assessment of imaging technology can be based on the source of the image. When the source is a television-related technology (video camera or tape player), the hardware for connection to computers is there for applications to be built. Some of the greatest interest in applications comes from the computer-based training community wishing to integrate scanned photographs and full-motion camera input with computer control. It is this community that is taking the lead on developing standards for data compression (see Section 9) for use in both data storage and data exchange.

For computer-generated simulations, on the other hand, the state of the art is improving at about the same rate that computing power is. Today's newest video graphics processors, which are board-level systems that plug into personal computers and workstations, include

in hardware much of the work done previously in software, such as three-dimensional rotation of solid objects.

In the automated data processing (ADP) community, imaging technology is also being driven by records management issues. Imaging technology is seen as a mechanism for getting paper records into a more compact form. Recent advertisements for CD-ROM systems, for example, claim that one CD can hold the images of all the documents in a full filing cabinet. This is spawning an industry in such devices as high-speed scanners for both color and black and white, and CD-ROM juke boxes. Again, with respect to data exchange, no special technology is being used yet. For data stored on CD-ROMs, the medium of exchange is the physical shipment of the discs.

One technology that has only very recently made the transition from "future" to "state of the art" is high-definition television (HDTV). The October 1989 issue of *IEEE Spectrum* contains an excellent overview in two articles of the state of development of HDTV [Jurgen 1989; Tannas 1989]. To summarize briefly:

- a. In Japan, daily broadcasts via satellite are underway on a limited basis (one hour a day in 1989, possibly up to eight hours per day by now). Viewing sites are in public areas since the equipment is still in prototype and very expensive.
- b. In Europe experimental broadcasts via satellite are scheduled to begin in 1991, with full service by 1995.
- c. In the U. S. HDTV broadcasting has been delayed by the lack of a terrestrial transmission standard (neither Japan nor Europe is attempting to develop such a standard). The Federal Communications Commission (FCC) has tentatively decided that any terrestrial transmission system must be receivable by current National Television System Committee (NTSC) receivers. The selection of a standard is not likely before 1992, with broadcasting not likely before 1995.
- d. In digital base-band, HDTV has about 5.7 times as many pixels as the NTSC image, while in the compressed broadcast form now used in Japan, it has about 4.9 times as many pixels as transmitted NTSC.
- e. The terrestrial broadcasting industry in the US and Japan has developed an alternative to HDTV called extended-definition television (EDTV) which is NTSC compatible. EDTV broadcasting was scheduled to begin in Japan in the

fall of 1989. EDTV receivers were on the market at that time for \$3000 for a 27-in. set.

Alexander [1990] describes some recent events in the U.S. HDTV industry. "In June, General Instruments Corp. stunned rival HDTV proponents when it unexpectedly submitted a completely digital HDTV system for Federal Communications Commission evaluation.... Two weeks ago [in November 1990], the FCC said it would begin testing HDTV systems in April 1991 as a preliminary to deciding on a standard.... The FCC said it planned to decide on an HDTV standard in the spring of 1993." Alexander quotes Sidney Topol, a fellow at the John F. Kennedy School of Government, as saying that the notion that Japan and Europe are ahead of the U.S. in HDTV is untrue. The HDTV picture is chaos and confusion in Europe, with as many differences as there are national boundaries. In Japan, where HDTV was launched earlier this year, the technology is not economically sound, in part because the televisions cost several thousand dollars. Alexander also quotes Stephen Weinstein of Bellcore as saying that HDTV is the "foundation technology for a wide range of industrial, professional and educational applications.... We'll see HDTV on workstations before home TV sets." That information was recently updated in a conversation with Mr. William Hassinger of the Federal Communications Commission. He stated that the evaluation actually started in July 1991, to be completed in 1992, with a standards decision due the second quarter of 1993. Some interim decisions are expected along the way. He also stated that the U.S. is taking a primarily digital approach, and that the Japanese are likely to reconsider their analog approach once the U.S. has issued its standard. Similarly, he claimed, the Europeans are having difficulty with their system, and he expects them to take a long look at the U.S. standard when it is available.

A.7.3 Standards

A.7.3.1 Current Standards

The only current standards related to video imaging are those adopted from the television industry such as RGB, NTSC, etc., and those derived from the graphics community such as GKS.

A.7.3.2 Current Standards Activities

For information on data compression standards related to photographs and television, see Section A.9

There is apparently no interest yet in the real-time simulation community in the development of standards beyond what is already available from the television industry. In fact, the attitude seems to be that the available computing power is not yet adequate to support standards. A recent article in *Computer Design* [Child 1990] quotes industry representatives as saying, "If you're in real-time imaging applications such as flight simulation, you've got to squeeze out every bit of performance that you possibly can," and "When you tell customers, 'Here are the trade-offs: You can either have high-speed performance or you can have standards,' high performance tends to overshadow standards."

Nevertheless, the Interactive Multimedia Association has supported the development of MIL-STD 1379D, *Military Training Programs*, which contains sections related to video standards. This standard is in final draft form at this time. Also, Microsoft Corporation has developed a specification for multimedia extensions to their Windows product that has been endorsed by IBM, Tandy, Fujitsu America and Zenith Data Systems [Krohn 1990]. The intention is clearly to create a *de facto* standard.

At least four types of standards are needed for HDTV: studio, exchange, including recording and network transmission, emission (broadcast), and display. In none of these areas is a single international standard likely. For example, in Japan, the number of scan lines per frame is likely to be 1125, with 60 frames per second; in Europe the standard is 1250 lines and 50 frames per second; and in the U.S. the total number of lines and the field rate have not yet been agreed upon. As noted previously, the FCC recently announced that it intends to issue HDTV standards in 1993.

A.8 AUDIO

A.8.1 CCIS Requirements

Integrated voice technology is identified as a requirement in [NIS ROC, 1983, 5]. Requirements for linking audio and computing systems in future CCISs will come from two directions. First, the desire to share scarce communications resources will create pres-

sure for services like Integrated Services Digital Network (ISDN). Second, once the ability to support voice becomes available, new applications for the technology will be demanded, such as voice mail for command and intelligence functions, multimedia documents for training and maintenance manuals, and computer-generated speech, particularly for "eyes-on" situations.

A.8.2 Related Technology

A.8.2.1 State of the Practice

The predominant application of digital audio today is in voice response systems. These are the systems where the computer 'talks' to a telephone caller while the caller responds using the buttons on a touch-tone telephone. Voice mail applications have been in use for several years, but most are only digital answering machines. At least one, the Wang product, treats voice mail as an extension of its electronic mail, permitting a message to be recorded and transmitted in digital form to another user's mailbox. The absence of standards for this application has hampered the spread of this kind of voice mail. The Wang system, for example, can only send voice mail messages to another Wang system. Speech recognition systems are widely available for limited application to situations where the speech can be discrete (generally at least one quarter second gaps between words) and the vocabularies are limited (typically one thousand words or less). Speech recognition does not involve the exchange of digitized audio.

A.8.2.2 State of the Art

ISDN services are now available in some areas of the world. ISDN seeks to combine audio, video and data transmission on a single system using a mechanism that first digitizes the audio and video. At the moment, the service is limited to a number of disconnected areas since most of the long distance trunk service has not been converted. In the U.S. the major long distance carriers are promising national service within a few years. Also, at present, most of the service available is for dedicated lines. Switched service is also promised in the near future. The use of ISDN is largely for the purpose of reducing line charges by big users of communications services by sharing the same lines between telephone and data usage. Here, the audio and data traffic are merged at one end and separated at the other.

There is no real integration of the two. Some experiments have been conducted with the transmission of multimedia documents in which the audio and data are integrated.

A.8.3 Standards

A.8.3.1 Current Standards

Standards related to ISDN are described in Appendix C.

A.8.3.2 Current Standards Activities

Future versions of MIL-STD-1379 (see A.7.3.1, Current Standards Activities for Video) is expected to include standards for digital audio. Work in this area is in progress by the Interactive Multimedia Association. In addition, there is a standard audio encoding method CCITT G.721.

A.9 DATA COMPRESSION

A.9.1 CCIS Requirements

The requirements for data compression in a CCIS stem from the need to exchange some of the preceding categories of data, where the volume of data is very large. Without data compression technology, the amount of data to be transmitted from one CCIS to another would exceed the communications capacity. In that sense, data compression is not a CCIS requirement; it is an enabling technology for meeting CCIS requirements. It is discussed in this appendix because the data compression technology and standards work has direct impact on several of the data categories described here. Two examples of data that fit this description are:

- a. At a scale of 1:1,000,000, a digitized map of the world requires thirty CD-ROMs. The Army wants maps that are 1:250,000 and 1:50,000. A 1:50,000 scale map of just the land portions of the world would be about 130 times bigger, assuming the same degree of clutter.

- b. The state-of-the-art standard for color video uses about 25M bytes per picture. Full motion video uses thirty pictures per second, or 750M bytes per second, most of a gigabyte link. It would take five hours to transmit one picture over a 9.6kb transmission line. These numbers are for an NTSC-compatible picture; for HDTV, they all go up by about 5.7, depending on which HDTV standard is ultimately adopted for the U.S.

A.9.2 Related Technology

A.9.2.1 State of the Practice

The desire to meet the above requirements has lead to a renewed interest in data compression as a means of storing and transmitting these massive data structures. In the past, data compression was viewed as a software problem, and the advantages of reduced storage and transmission size were offset by the slowness of the compression and decompression algorithms. Also, decreases in storage and transmission costs always seem to overtake the compression technology, making its application unnecessary. Today, the substantial reductions in the cost of custom chip design have led a few manufacturers to start selling data compression and decompression chips that operate at very high speeds. Very little actual use has been made of these chips yet, but a standard for the compression/decompression algorithms is emerging that promises to make their use much more practical in the near future.

A.9.2.2 State of the Art

The algorithms in use today are advertised to achieve data compression ratios of 20:1. At the moment, data compression is being used largely in connection with data storage.

A.9.2.3 Future Research

The companies that have developed chips claim that they are working on compression ratios of 200:1. Another company is claiming that compression techniques based on fractals can achieve ratios of 500:1 or higher.

At the same time, storage densities are growing substantially. A recent press release reported two companies have decided to produce a new storage system with a capacity of 7000 terabytes, or the equivalent of 50 million reels of magnetic tape (2400 feet at 6250 bpi) [Shapiro 1990]. Research is also underway on gigabit per second networks. Consequently, it is possible that today's anticipated needs for data compression will evaporate in the face of this advancing technology.

A.9.3 Standards

A.9.3.1 Current Standards

There are two data compression standards already approved, video Codec for Audiovisual Services at $p \times 64$ kbit/s ($p \times 64$) CCITT H.261 and Digital processing of video signals—Video coder/decoder for audiovisual services at 56 to 1,536 kbits/s ($p \times 56$) ANSI. The H.261 recommendation (all CCITT standards are called recommendations) describes a family of compression standards, one for every integer value of p , to provide a range of performance characteristics. Intended applications are for videophone (using $p = 1$ or 2) and video conferencing (using $p \geq 6$). The CCITT recommendation was approved in late 1990. A slightly modified version was adopted by ANSI for use in North America. A few products available in the US currently use the ANSI version of the standard. Many more are expected to be announced soon. In the US, the ANSI version is becoming a de facto standard for videophone and video conferencing applications. Therefore, interoperability between ANSI and CCITT systems is likely to be a problem for international situations.

A.9.3.2 Current Standards Activities

The Joint Photographic Experts Group (JPEG), a joint project of ISO and CCITT, has issued a proposed standard commonly referred to as the JPEG standard, but officially ISO 10918. JPEG was developed to support compression of full color still images for fax transmission. It is being adopted for storage and transmission of images independently from fax, and, since it is a symmetrical compression algorithm, some vendors are also using it for full motion applications. Since JPEG is a lossy compression technique (that is, on decompression not all data is recovered), it is not suitable for applications where total fidelity with the original is required. The technology on which the JPEG standard is based is simple math-

ematics, the Discrete Cosine Transform. Its use for two dimensional data compression is well understood. The ability to implement this technology in a single chip operating at real time speeds (30-60 frames per second) is relatively new, but does not require innovative technology. JPEG is expected to be approved as a full standard in 1991, and chip and board level products based on draft versions of the standard are already available and in use. Recent changes to the standard that have occurred as part of the review process have all been minor ones. Vendors have had enough confidence in the standard to start shipping products since the spring of 1990. Since then, many vendors have started incorporating JPEG in products being shipped and planned for release in the very near term. It will be used for a variety of imaging applications, but not all.

A second proposal is under development by the Moving Picture Experts Group (MPEG). MPEG is an asymmetrical data compression standard that combines DCT compression within a single frame with additional compression that eliminates redundancy between frames. Its intended use is with motion video, but since it is asymmetrical, it is very expensive to use today in real-time applications, making it most suitable to publishing applications. MPEG is in draft form, with full approval expected sometime in 1992. Initial products should be out within the next few months. Single chip implementations that could have a significant impact on usage have been promised before the end of 1992. It is likely to be used heavily for some applications.

A third proposal is under development by the Joint Bilevel Imaging Group (JBIG). JBIG is being developed for improving the quality of compressed images that have only two colors, like black and white. It is expected to be used for compressing black and white photos and images of text documents. It is expected to find heavy usage in fax transmissions since it produces superior quality images for the same bandwidth as Group 3 and 4 fax standards. JBIG is a recent development, still in early draft stages. To date, no products using JBIG are known to have been produced.

A potential de facto standard, called Digital Video Interactive (DVI), uses a proprietary compression scheme, but is backed by Intel Corporation, IBM, and AT&T. Already, IBM and Intel are marketing DVI products for personal computers.

A.10 ISSUES

Looking toward an architecture for the time period of 1995-2010, the issues related to data exchange can be grouped into a few main categories (it is not claimed that either this list or any of the sublists are complete):

- a. Standards. For the near term, most of what could reasonably be built into systems starting in 1995 is either in place or under development. Some standards, particularly for audio and multimedia documents, are not very close to being complete, but the demand is rising very rapidly within the mainstream of computing activity. The probability that the standards will be in place by 1995 is quite high. Perhaps the one area of greatest uncertainty is HDTV. The amount of research in this area is sufficiently high that products should be available by 1995, but the political issues being raised, particularly within the U.S. surrounding its adoption may delay standardization and commercial availability beyond that time. Also, it may be some time after 1995 before HDTV is considered cost effective.
- b. Replacing formatted messages with database exchanges as the primary means of data exchange. This issue has a number of important problems that need to be resolved, including the following:
 - (1) How quickly will users adapt to electronic "documents" and the replacement of formatted messages by database exchanges?
 - (2) Which formatted messages, if any, should never be replaced by database transfers?
 - (3) What is the most appropriate approach to replication and how to manage it?
 - (4) What concept of data ownership should be employed?
 - (5) How will the common conceptual schema that crosses organizational boundaries be developed?
- c. Multimedia. This issue also has a number of important problems that need to be resolved, including the following:
 - (1) How quickly will users demand multimedia capabilities in operational information exchanges, such as for situation reporting, orders dissemination and teleconferencing?

- (2) When will simulation outputs be added to operational information exchanges?
- (3) When will the communications infrastructure be upgraded sufficiently to permit the exchange of all the kinds of data that may be desired?
- (4) When will database systems support objects the size of video segments?

There are technical, political and cultural problems that contribute to delays in accomplishing what appears to be only a matter of working out the details.

APPENDIX B —DATA MANAGEMENT

B.1 INTRODUCTION

Data management is concerned with the design, allocation, control, and maintenance of data in a system. Data management services must be provided for all data that support a CCIS such as text, numeric data, and complex objects such as formatted and unformatted documents, graphics, maps, meteorological data, video and audio (see Appendix A). This appendix discusses CCIS data management technologies expected to be available to make that possible.

Four steps were taken in developing this appendix. First, an understanding of CCIS data management needs was acquired from examining documents that discuss the Worldwide Military Command and Control System (WWMCCS) requirements, goals, and deficiencies [WAM DCP 1989]. Examining WWMCCS provides many lessons in the difficulties of improving an existing complex command and control system. Next, an assessment was performed of current and future hardware and software technologies applicable to CCIS data management. In parallel, an assessment of current and future software standards applicable to CCIS data management was developed. Finally, underlying issues that relate to CCIS data management were identified and discussed.

This appendix addresses technology and standards for data management that can be expected to mature within 5 to 10 years (the near-term) and those that may be applicable in 10 to 20 years (the long-term). It includes the management of complex objects. Data management as part of the CCIS is considered at all echelons and in all functional areas. While most of the specific statements of requirements have come from documents related to WWMCCS, the technical discussion presented here is not limited to solutions to the specific problems of WWMCCS.

The remainder of this appendix is organized into the following sections: CCIS data management needs, hardware and software technology assessment, standards assessment, and issues.

B.2 CCIS DATA MANAGEMENT NEEDS

CCIS data includes: status information (about forces, supplies, and equipment), intelligence information, plans, significant events, execution reports, environmental data such as weather observations and forecasts, the status of the CCIS and communications network, guidance from higher command authorities, and reference data such as maps, treaties and agreements, maintenance manuals, and training materials. Data management services must support the storage, retrieval and dissemination of this data securely and flexibly in a timely manner, ensuring that CCIS users are kept up-to-date in their individual areas of concern.

Using the terminology of relational database management systems, the database of the global CCIS community is partitioned and partially replicated. It is partitioned horizontally, meaning the same kind of data is held at multiple locations. For example, targets might logically represent one kind of data (or perhaps a few distinct kinds), but all units engaged in a war or crisis situation would probably maintain their own target lists. The database is also partitioned vertically, meaning not all information related to the same object is necessarily stored in the any one location. It is partially replicated because some, but not all, the data held at one location is shared with some, but not all other locations.

In general, across the entire military command structure, there may be thousands of CCIS nodes, where each node needs access to some of the information maintained by each of the other nodes that is organizationally close to it. The notion of organizational closeness is expressed, today, in terms of information exchange requirements which are determined largely by doctrine, but in some cases by individual commanders. The worldwide set of information exchange relationships forms a network that includes every CCIS node, but it is not intended to provide access from every node to all data everywhere in the network. These information exchange requirements are not static. Changes in organizational closeness occur as a result of shifts in military structure during peacetime, crisis and war. DoD is currently undergoing a number of organizational realignments due to changes in the world political order and in response to budget cuts. During the recent Persian Gulf crisis, units unaccustomed to operating together were deployed to the region and expected to form close working relationships. In the course of a war, attrition and the shifting points of concentration can bring about the same results. In addition, some of the information exchange relationships are with systems that are not CCISs. Wherever one draws the line on what is a CCIS and what is not, at the boundaries, there are information exchange requirements.

Also, because of the large number of nodes involved representing a diverse community of users and a wide variety of systems, the overall database system must be expected to be heterogeneous. That is, it will make use of hardware and software representing a variety of vendors, supporting a variety of data models and, even for a single vendor's product, a variety of versions of everything including standards.

This description of the CCIS database is intended to simply describe the world as it is, without making any inferences regarding how this global database should be managed. How the data should be managed is a function of the operational requirements and the available technology. As described above, the operational requirements call for a data management system that is worldwide in scope; that has local management control to provide the flexibility needed to support dynamic exchange requirements; and that provides support for the integration of a wide variety of kinds of data (numeric, text, images, etc.)

The data management system must also have the characteristics required of the CCIS in general. Some of the key ones that directly influence data management are security, survivability, interoperability, and flexibility. The description of the available technology suitable for meeting these requirements is the subject of the remainder of this appendix.

B.3 HARDWARE AND SOFTWARE TECHNOLOGY ASSESSMENT

This section discusses near-term and long-term hardware and software technologies that are expected to be available for CCIS data management.

B.3.1 Near-term Hardware Technologies

Computer hardware technologies that affect data management may be divided into those that are general and those that are specific to data management.

B.3.1.1 General Hardware Improvements

Data management will be affected by near-term hardware improvements in processors, memory, storage, and networks. Hardware processing power will continue to increase rapidly and the cost of main and disk memory to decrease. The most important technologies are parallel processing, parallel input/output (I/O), and fiber optic networks. Parallel pro-

cessing and parallel I/O can be expected to become available for general purpose workstations in the near future. Such availability will remove a major distinction between workstations and mainframes and allow great capability for data processing. Several scalable database prototypes have shown that almost linear speedup may be obtained via use of parallel processing (no shared resource to limit the implementation scale) [Boral 1990; DeWitt 1990]. One survey predicts the following trends for database technologies [Stonebraker 1989].

- a. All database systems will be distributed in a few years.
- b. Since conventional machines are improving so rapidly, "it seems unlikely that a hardware database machine vendor can develop cost-effective CPU's."
- c. Most relational database systems will increasingly improve transaction rates.
- d. The large decrease in main memory prices will allow some databases to reside entirely in main memory.
- e. New storage architectures will become available, such as optical disk devices or disk farms of inexpensive small drives.

Forecasting the technological changes in a field as rapidly changing as computer hardware is susceptible to a wide margin of error. An estimate of computer chip technology in the year 2000 based on an extrapolation from the change over the last 10 years is given in Table B-1 [Hall 1990].

Table B-1. Microprocessor Technology Forecast

Year:		1980	1990	Extrapolation	2000
Processor	Pins	64	256	x4	1k
	Clock	8 MHz	32 MHz	x4	128 MHz
	Devices	50k	1M	x20	20M
	MIPS	1	20	x20	400
Memory (DRAM)	Bits/Chip	16k	4	x256	1G
	Speed	240 ns	80 ns	+3	25 ns

An estimate, more optimistic than indicated in the table, by engineers at Intel suggested that microprocessor chips in 2000 would be 250 MHz, one square inch, with a 2MB on-chip cache, allowing room for over 2000 pins [Intel 1989]. Similar dramatic advances are expected in storage technology and I/O. Storage systems containing up to 7000 terabytes

are under development. Microprocessors have had better price/performance ratios than mainframes and supercomputers for some time, but for a variety of reasons have not been able to "scale up" to replace them. One reason, the lack of high bandwidth I/O, seems likely to disappear in the near future as wider data busses, arrays of inexpensive disks with large cache memories, and multiprocessor, multidisk systems allow an aggregate I/O bandwidth approaching that of a supercomputer.

The development of networks capable of handling bandwidth measured in the gigabytes will have a significant impact on data management because it will make available a whole new range of objects within the information system. These objects, formatted and unformatted documents, graphical data, maps, meteorological data, video, and audio, are discussed in Appendix A: Data Exchange. The development of high-speed networks, which are a national priority [OSTP 1989] and a subject of much commercial interest, is expected to proceed so rapidly that the exchange of large objects will be practical in the next 5 years (for detailed discussion, see Appendix C: Network Services). Data management for the CCIS needs to identify, manage, display, and use effectively the objects most needed by the system.

B.3.1.2 Data Management Specific Hardware Improvements

A computer designed specifically to support data management is called a *database machine*. Some typical approaches are the use of a proprietary "smart bus" to retrieve data (Teradata's YNET), microcoding to accelerate queries (used by Britton-Lee several years ago), massively parallel processors (Connection Machine or MasPar), or in-memory database systems with very large memory.

The history of database machines shows that they were initially received enthusiastically [Hurson 1989], but have since been regarded more coolly [SIGMOD 1989; Stonebraker 1989]. This lack of enthusiasm is not due to an inability of database machines to meet their objectives—Teradata offers a commercial database machine with excellent access to distributed disks, data "mirroring" (replicated transactions to a backup disk) for reliability, and useful features for security. Rather, general purpose processors have improved so rapidly that it is hard for a proprietary hardware architecture to keep up. It is unlikely that a single company could continue improving its hardware, operating system, and database management software enough to stay competitive.

In addition, a software database management system (DBMS) offers more flexibility than a database machine. For example, the Gamma system, running on 17 VAX 11/750 processors connected by a local area network, has retrieval times comparable to Teradata, but with much better performance on complicated queries [DeWitt 1990]. Also, a software-only DBMS more directly supports the requirements for survivability needed in the CCIS environment by being distributed over a larger set of nodes than a database machine.

B.3.2 Near-term Software Technologies

Software technologies that affect data management may be divided, like hardware technologies into those that are general and those specific to data management. General improvements in software data management technologies can be expected. Most of the improvement in data management will come from specific data management software.

B.3.2.1 General Software Improvements

The improvement in network management software and inter-networking that takes advantage of the higher speed network hardware will mean that data will support data management by making the distribution of more data possible, easier, and faster. Access to data and data manipulation functions will likely become almost entirely location transparent, depending only on the authority of the user to perform the task.

The object-oriented paradigm is becoming prevalent in many aspects of computing because of the natural way it allows one to model interactions among system components. Since database systems are expected to include object-oriented capabilities (see below), there will probably be a more uniform user interface to the different parts of the CCIS. In addition, the operating system interface and database interface will both be window oriented, all of which will combine to make it easier to build systems that present data to users and accept data from users in more meaningful ways.

Other application programs that partially support the database function are multimedia authoring systems and hypertext systems. Such systems make it easier for the user to build large, complex objects that involve not only ordinary data, but relationships among parts of the database and large, non-decomposable objects. Similarly, information retrieval technol-

ogy that supports full text searches is being seen increasingly as something that needs to be integrated with the overall data management function.

In addition, many COTS applications, such as spreadsheet processors and word processors, provide interfaces to database management systems. Such integration makes both the application and the DBMS more useful. Today, many of these are tightly coupled in that the application only works with one brand of DBMS, but the trend is clearly in the direction of using standard interfaces, most notably SQL, to provide links to any DBMS that conforms with the standard.

B.3.2.2 Data Management Specific Software Improvements

A great deal of work is now going on to introduce object-oriented characteristics into databases [Stonebraker 1989; SIGMOD 1989; UCB 1990; Atkinson 1989]. The argument can be made that an object-oriented model is a much more natural way to model the real-world. For example, an object-oriented database system could store a briefing on a developing crisis that included text, maps, formatted messages, video from news reports on the situation, and formatted plans and orders. There is disagreement at present between those who think that an entirely different model is needed for object-oriented databases and those who think that relational database systems can be extended, but the two views may not be very far apart in practice. With the exception of zealots in both camps, most experts agree that there is a need to support multiple data models because different kinds of data have different requirements. We expect future databases to have good capabilities in traditional database functions (e.g., SQL) and in object-oriented capabilities.

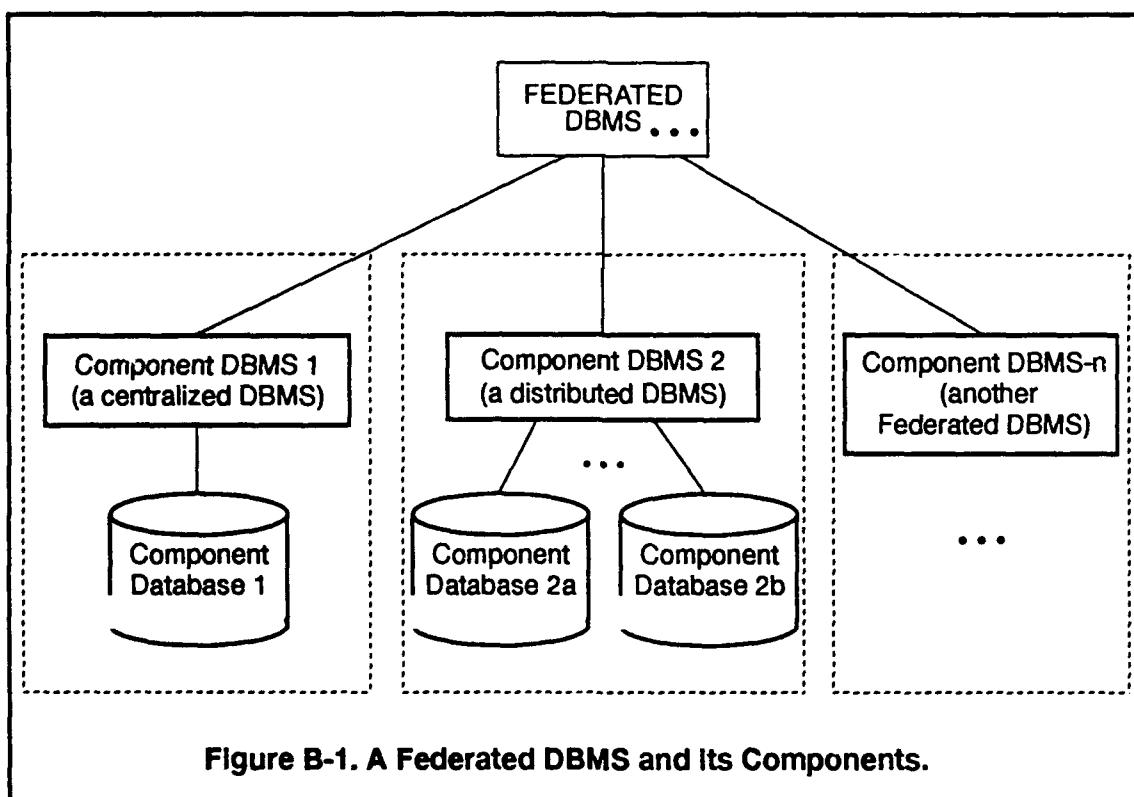
Large rule-based systems are currently being developed. Originally written in artificial intelligence languages, these systems are now using relational database systems to allow larger capacities and ease of maintenance. The developers of these systems have found heuristics to rapidly search databases as large as several Gigabytes and to incrementally update them to efficiently maintain the interrelationships in the data [Cohen 1990].

Distributed database systems were cited by the Laguna Beach report of the National Science Foundation (NSF) Workshop on the Future of Database Systems Research [Silberschatz 1990] as a significant accomplishment of database research in the last two decades. The report notes that "all the major DBMS vendors are presently commercializing distributed DBMS technology" [Silberschatz 1990, 8]. The areas of greatest promise in the future

are thought to be next-generation database applications and heterogeneous, distributed databases.

The Laguna Beach report mentions the challenges of very large databases (e.g., satellite data, engineering design data, intelligence data) and the need to find patterns in these databases. Large knowledge bases of rules will be created. New kinds of objects, as discussed previously, will be managed. The report identifies the need for parallelism and algorithms that scale up and for long duration transactions.

One technology that seems particularly well suited to the CCIS situation because it very closely models the description of the global CCIS database given above is the federated database. A federated database system consists of a collection of cooperating but autonomous component databases that are integrated to various degrees as shown in Figure B-1.



The diagram illustrates this approach, where multiple autonomous, distributed, and heterogeneous DBMSs are connected to allow access to data [Sheeth 1990]. Individual DBMSs retain site autonomy (e.g., what data is being managed, query language used, access control, semantic interpretation of the data) yet participate in a federation to allow partial and

controlled sharing of their data. The DBMSs in the federation may be either tightly coupled, where management of the federation is centralized, or loosely coupled, where the local DBMS must manage its federation responsibility. The tightly coupled federation can serve as a model for a mechanism for integrating multiple data models within the same CCIS node, as well as integrating centrally managed multiple DBMSs on a common LAN. The loosely coupled federation can serve as the model for the integration of organizationally independent databases where a high degree of autonomy is needed to support organizational flexibility requirements or a commander's directions for access controls. In the short term, not all these forms of integration will be available. The technology is relatively new, but is receiving a significant amount of attention. Some products already support some aspects of federation, with promises of more support on the way.

B.3.3 Long-Term Hardware Technologies

Some of the longer range technologies that may affect data management are neural networks and optical computing. Neural networks have proven their effectiveness at pattern recognition tasks. They might become a useful tool in queries and updates of databases containing nonstandard objects, such as graphics. There is also research going on in relational databases implemented on massively parallel processors using a parallel neural network architecture [Lydic 1990]. Optical computing may eventually provide great speed at low power. For example, a digital optical processor has been demonstrated by AT&T Bell Labs that operates at 1 million cycles per second but is believed potentially capable of 1 billion operations per second using a switching energy of 1 picojoule [OE 1990].

B.3.4 Long-Term Software Technologies

In its long range view of future database technologies, the Laguna Beach report also envisages "the existence of a single, world-wide database system, from which users can obtain information on any topic covered by data made available by purveyors, and on which business can be transacted in a uniform way." Similarly, the ISAT Summer Study [1990] proposes the creation of a Defense Information Infrastructure that provides information in a way similar to the telephone or electric power system. This requires that the various database systems be able to adapt to the disparate structures, so that they compensate for variations in their semantics. Each system would need to represent the semantics of its

information in a standard way to inform those who queried it. Some of the problems in doing this are incompleteness and inconsistency, name services, security, and transaction management.

Lenat's "CYC" effort [Lenat 1989] aims to capture "common sense" knowledge of the world as a base on which to build the semantic knowledge of many areas. Lenat hopes to create a standard product that will become as normal a part of computers as a spell check program. In the long run, the automation of knowledge may lead to greater rigor, clarity, and ease of design in many fields. This and other efforts in natural language understanding hold promise for the user being able to carry out a dialog with the data management system.

B.4 STANDARDS ASSESSMENT

This section of the appendix discusses ANSI and International Standards Organization (ISO) standardization efforts relevant to CCIS data management. Primary emphasis in this area has focused on common data dictionary systems, languages to access DBMSs, and remote access of databases. Other areas of activity include the definition of a framework for data management, standards for open distributed processing, and distributed transaction processing. Each of these areas will be addressed in this section of the appendix.

B.4.1 Reference Model of Data Management

The Reference Model of Data Management (RMDM) establishes a framework for coordinating the development of existing and future standards for the management of persistent data in information systems. It defines common terminology and concepts pertinent to all data held within information systems. Such concepts are used to define more specifically the services provided by particular data management components, such as database management systems or data dictionary systems. The definition of such related services serves to identify interfaces which may be the subject of future standardization.

The RMDM does not specify services and protocols for data management. The standard is neither an implementation specification for systems, nor a basis for appraising the conformance of implementation.

The scope of the RMDM includes processes which are concerned with handling persistent data and their interaction with processes particular to the requirements of a specific

information system. This includes common data management services such as those required to define, store, retrieve, update, maintain, backup, restore, and communicate application and dictionary data. The RMDM includes consideration of standards for the management of data located on one or more computer systems, including services for distributed database management. It does not include common services normally provided by an operating system including those processes which are concerned with specific types of physical storage devices, specific techniques for storing data and specific details of communications and human computer interfaces. It defines services provided at an interface; it does not impose limitations on how processes are implemented.

The standard which defines the RMDM seeks to standardize much of the terminology related to the structure of a data management system. Most of the terms it defines are consistent with common usage. One term which is likely to cause confusion unless its standard definition is understood is 'processor.' The processor plays a central role in the RMDM, but it does not represent any relationship to hardware or software. Most of the entities described as processors are most commonly implemented as software, but the standard does not require that. The RMDM defines a generic model of data management as shown in Figure B-2. The components shown are intended to be abstract classes. In the figure, the

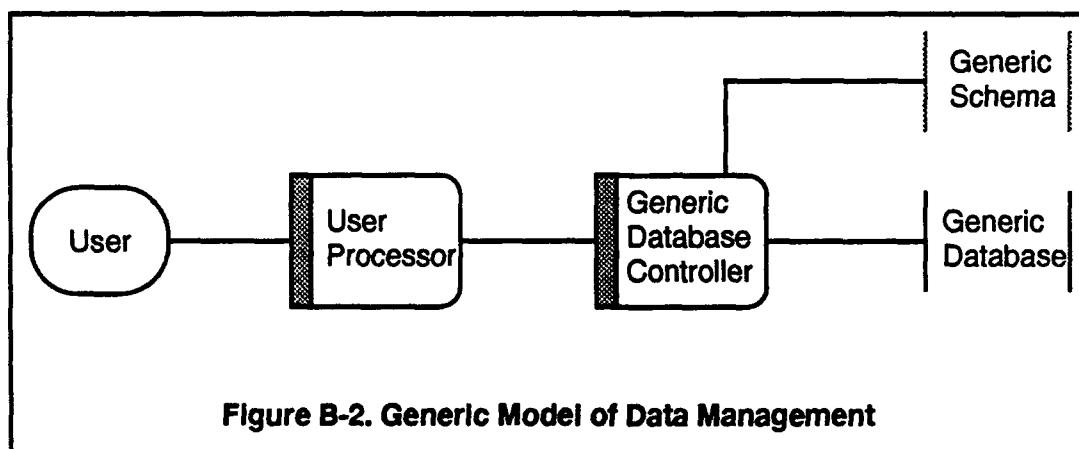


Figure B-2. Generic Model of Data Management

boxes with two rounded corners and one shaded end are processors, with the shaded end representing an interface which is a candidate for standardization. A number of specializations of this generic model are derived by decomposing one or more of the generic classes into other classes. Specific specializations show how to support multiple simultaneous users, multiple databases, distributed data management, export/import files, and access control.

B.4.2 Information Resource Dictionary System

Database management systems provide mechanisms to define and manipulate data stored by an organization. Traditionally, these DBMSs also provide facilities for accessing stored descriptions of data. Such descriptions typically found in a DBMS include, for example, lists of privileges associated with the data, and all tables stored in a DBMS. Facilities for DBMS descriptive data are called data dictionaries, data directories, system catalogs, and schema manipulation languages.

The purpose of an Information Resources Dictionary System (IRDS) is to expand and generalize the descriptive facility to incorporate not only data managed by a DBMS, but all information associated with an organization. Information managed by an IRDS includes that associated with application programs, files, users, and hardware. Proper development and maintenance of an IRDS could assist in answering the following types of questions:

- a. How many files and programs will be affected by changing a basic data element?
- b. In a distributed environment, what clients are using what resources (e.g., files, programs, data elements)?
- c. What computing equipment (i.e., hardware and software) exists in the organization, who uses it, and what is it used for?
- d. How can a consistent view of local and distributed data be implemented and effectively managed?

B.4.2.1 IRDS Standardization

In 1980, ANSI created the X3H4 committee to develop a standard IRDS. Concurrently, the National Bureau of Standards (now NIST) initiated an effort to develop a Federal Information Processing Standard (FIPS) for data dictionary systems. These efforts merged in 1983, resulting in an ANSI IRDS standard in 1988. NIST has adopted ANSI IRDS and provided procurement guidance to federal agencies in the form of FIPS PUB 156.

In 1984, ISO formed TC97/SC21/WG3 to develop an international IRDS standard. The ANSI IRDS specification was used as the ISO baseline, and the two specifications initially progressed concurrently. However, in 1988 the ANSI and ISO specifications diverged. The divergence centered on the selection of the underlying IRDS data model. ANSI IRDS

assumes the use of an Entity-Relationship Model, that provides an underlying IRDS foundation. ISO has adopted an SQL-like data model that can be readily implemented using current commercial relational DBMSs.

B.4.2.2 Outlook

The purpose of an IRDS is to provide effective means for defining, manipulating, and updating data within an organization. IRDS's precursors have existed for many years in the form of data dictionaries, directories, system catalogs, etc. However, these systems typically focused only on data stored in a DBMS, not all data pertinent to an organization. During the last 10 years, there has been a push to develop such systems, and the recent ANSI standard IRDS specification provides such facilities. For command and control systems, it is critical that data be consistently defined and distributed appropriately (e.g., centrally, locally, federated), that the existence and location of data is known by those who require such information, and that the data is maintained effectively. Appropriate use of an IRDS, either an implementation of the ANSI standard IRDS or an IRDS-like COTS product, would facilitate each of these concerns. Implementations of the ANSI IRDS specification have not yet been developed. Although the NIST has issued FIPS PUB 156 requiring ANSI IRDS usage, vendor commitment to developing such implementations cannot be assessed at this time. If quality implementations of ANSI IRDS are produced in the next several years, they would certainly be appropriate for use within a CCIS environment.

In March 1991, an 18-month transition period ends for FIPS PUB 156, and ANSI IRDS becomes the recommended information resource dictionary and data dictionary/directory for federal use. No commercial implementations of ANSI IRDS exist today, partly because there is no method to measure and test conforming implementations. NIST expects to produce an IRDS conformance test system in early 1991. Vendor commitment to the ANSI IRDS standard cannot be adequately estimated until the availability and use of the conformance test system. Convergence of the current ANSI and ISO IRDS efforts is extremely unlikely. The ISO IRDS is currently at the Draft International Standard (DIS) stage, and will probably be finalized in two years. Thus, it is likely that ANSI and ISO will have separate standards for IRDS in the near-term. Both ANSI and ISO are exploring what the next generation standard IRDS will look like, but it will be at least 5-10 years before a second generation IRDS standard is developed.

B.4.3 Database Language SQL

The majority of activity in database standards over the last 10 years has been focused on SQL. Commercial DBMS vendors have been primarily producing relational database products with an SQL interface. Given the current and future impact of SQL upon the data management standards community, commercial vendors, and the federal government users, much attention will be given to SQL in this appendix.

B.4.3.1 SQL Standardization

In 1982, the ANSI X3H2 Database Committee began work on developing a standard relational database language. In 1986, X3H2 produced an ANSI standard SQL, referred now to as SQL1, which provided facilities for defining, manipulating, and controlling data in a relational database [ISO 9075, 1986]. The SQL1 standard has since been revised to incorporate support for integrity enhancement, and is now called Database Language SQL with Integrity Enhancements [ISO 9075, 1989]. NIST has adopted this ANSI SQL and provided procurement guidance to federal agencies in the form of FIPS PUB 127-1.

B.4.3.2 Interface Considerations

The SQL1 standard does not provide syntax and semantics for dynamic execution. Thus, the standard does not address interactive queries to a DBMS. Instead, the standard defines two methods of interfacing (i.e., binding) a host programming language with SQL. The first method uses a “module language” that enables an application written in a host programming language (i.e., COBOL, FORTRAN, Pascal, and PL/I) to execute a previously compiled SQL module. Data can be passed to and from the application and the DBMS through the SQL module. The second method is referred to as language embedding. In this approach, SQL commands are embedded within an application program. A preprocessor is then used to extract the SQL commands from the application program and generate the appropriate object code needed to access the DBMS.

There has been much activity outside of ANSI or ISO in developing an alternative method of interfacing Ada and SQL. This method, SQL Ada Module Extensions (SAME), is based on the SQL module language, but provides facilities to the application program for exploiting features of the Ada language, such as strong typing [Graham 1989]. The SAME

method of binding appears to be gaining support in both the SQL and Ada communities. CCIS environments that include Ada application programs manipulating SQL DBMSs must decide which type of interface method to use.

B.4.3.3 FIPS PUB 127-1

FIPS PUB 127-1, issued by NIST, adopts the SQL1 standard, an embedded language standard, and provides additional guidance and considerations for use in SQL procurements. Issues relating to FIPS PUB 127-1 are discussed in the following sections.

B.4.3.3.1 Provisions

FIPS SQL includes all provisions from ANSI X3.135-1989, Database Language SQL with Integrity Enhancement, and ANSI X3.169-1989, Database Language Embedded SQL, with the following exceptions:

- a. FIPS SQL does not recognize Level 1 ANSI SQL or partial conformance to just Data Definition Language (DDL) or Data Manipulation Language (DML). Conformance to Level 2 ANSI SQL is required. Most commercial implementations of SQL provide the Level 2 functionality.
- b. FIPS SQL does not include PL/I language bindings, since PL/I is not a FIPS programming language.
- c. FIPS SQL does not recognize conformance solely by “direct invocation of SQL data manipulation language statements” as specified in Section 3.4 of X3.135-1989, because that concept is not adequately specified in ANSI SQL and implementations cannot be tested for conformance. Conformance to FIPS SQL requires a Module Language or Embedded SQL interface to one or more FIPS programming languages.
- d. FIPS SQL includes a “FIPS Flagger” requirement.

B.4.3.3.2 Procurement Considerations

FIPS SQL includes various alternatives for interfacing to programming languages; specifies “integrity enhancement” as an optional component of the standard, and does not specify any minimum requirements for the size or number of occurrences of database constructs. FIPS PUB 127-1 also provides a set of special procurement considerations in order

to assist SQL procurements. In a procurement, references to FIPS PUB 127-1 have the following requirements:

- a. *Integrity Enhancement Feature.* Should indicate whether or not the “integrity enhancement” feature (an optional component of X3.135-1989) is required. Failure to make this indication means that the feature is not required.
- b. *Programming Language Interfaces.* Should indicate which programming languages (e.g., Ada, C, COBOL, FORTRAN, or Pascal) are to be supported for language interface. Failure to make this indication means that support for any one of these languages satisfies the FIPS SQL requirement.
- c. *Style of Language Interface.* Should indicate, for each programming language identified above, whether the language interface is to support Module Language, Embedded SQL, or both. Failure to make this indication means that support for any one interface satisfies the FIPS SQL requirement.
- d. *Interactive SQL.* Should indicate whether or not “direct invocation of SQL statements” is required and, if required, which SQL statements are to be directly invocable. Failure to make this indication means that direct invocation of SQL statements is not required. If no such minimum requirements are proposed, FIPS PUB 127-1 identifies minimum requirements for interactive SQL.
- e. *Sizing for Database Constructs.* Should indicate minimum requirements for the precision, size, or number of occurrences of database constructs. If no such minimum requirements are proposed, FIPS PUB 127-1 identifies minimum sizing requirements for various database constructs (e.g., length of an identifier, minimum data type precision values, etc.).
- f. *Character Data Values.* The set of character values for the character data type and the collating sequence of characters in SQL are both implementor defined. References to FIPS PUB 127-1 should indicate any additional character data requirements. If no character data requirements are proposed, support for representation of the 95-character graphic subset of ASCII in an implementor specific collating sequence is by default the minimum requirement.

B.4.3.3 FIPS Flagger

ANSI SQL allows an SQL implementation to provide additional facilities that are not specified in the standard. In order to assist in identifying such supersets, FIPS SQL requires

a conforming implementation to provide a "flagger" capability that allows identification of implementor extensions. The FIPS Flagger is intended to effect a static check of the SQL language. Any SQL language that violates the Format or Syntax Rules, except privilege enforcement rules, is an extension and must be flagged. The FIPS Flagger should be implemented as an option to be used by the application programmer.

B.4.3.3.4 NIST Conformance Testing

A suite of automated validation tests for SQL implementations has been developed by NIST. It is planned that an enhancement of this test suite will be the basis of a future certificate of validation offered to implementations claiming conformance to this standard. NIST has begun issuing registered validation reports based on successful conformance testing. It is expected that in early 1991 NIST will begin a formal testing service and will issue validation certificates.

B.4.3.4 Current SQL Standardization Activities

ANSI X3H2 is currently focusing on two projects: SQL2 [ISO DIS 9075.2] and SQL3. SQL2 provides greatly increased functionality to the current SQL, and SQL3 can be characterized as an SQL "wish-list". The draft SQL2 document is currently a draft standard, with acceptance in 1992. Fully conforming SQL2 implementations may take several years to appear after the SQL2 standard is finalized. The draft SQL2 working document supports:

- a. Referential integrity (e.g., cascades, set null/default)
- b. Domains
- c. Table operations (e.g., outer join, intersection, difference)
- d. Data types (e.g., date/time, variable character, national character sets, etc.)
- e. Enhanced functions (e.g., substring, concatenation, case (if/else))
- f. Schema manipulation language
- g. Access to schema table information
- h. Scroll cursors/interactive browse
- i. Enhanced transaction management (e.g., read/write)
- j. Increased orthogonality
- k. Temporary tables and views

1. Dynamic SQL.

The SQL2 document also specifies three levels of SQL2 conformance: entry, intermediate, and full. The entry level is essentially the ANSI X3.135-1989 subset of SQL2. The intermediate level includes features such as schema manipulation language, dynamic SQL, multiple module support, numerous functional enhancements, etc. The full level increases SQL2 orthogonality, named constraints, date, time and interval data types, scrolled cursors, additional support for referential integrity, etc.

ANSI X3H2 work on SQL3 is in a preliminary stage. Most of the features that were not accepted into SQL2 were placed into SQL3. Features such as triggers, assertions, user-defined data types, and object-oriented capabilities are currently in SQL3. It is unlikely that the SQL3 standard will be adopted prior to 1995. Conforming implementations of SQL3 will probably not appear prior to 2000.

B.4.3.5 Outlook

A major aspect of CCIS data management is the issue of accessing databases. There are two factors which indicate SQL will play an important role in CCIS data management. First, the commercial marketplace provides many SQL-based products. These products include not only DBMSs, but data dictionaries and directories, 4GLs, report writers, spreadsheets, etc., which are based upon SQL in some fashion. Thus, SQL provides a foundation for developing many tools for accessing a DBMS. A second factor is the commitment of the commercial marketplace to the ANSI SQL standard. Virtually every major DBMS vendor has a representative on the ANSI SQL committee, several DBMS vendors have produced or are about to produce NIST-conformant implementations of ANSI SQL, and there is much interest by vendors in pursuing the forthcoming SQL2 standard. It should also be noted that DBMS vendors are not producing new products which are based on the network or hierarchical data model. Vendors are keenly aware that SQL is and will continue to be the language of choice for accessing relational DBMSs for many years to come.

Several conforming implementations of FIPS PUB 127-1 exist today, based on NIST's SQL conformance test suite. As previously mentioned, vendor commitment to FIPS PUB 127-1 appears strong. The same level of commitment can be expected for SQL2. With standardization of SQL2 likely in 1992, conforming implementations should appear in the mid-

1990 timeframe. Thus, SQL2 will be the primary DBMS language for many years to come. At this time, the impact of SQL3 cannot be anticipated.

B.4.4 Remote Database Access

The goal of the Remote Database Access (RDA) effort is to allow interconnection of applications and database systems from different manufacturers, under different managements, of different levels of complexity, and exploiting different technologies. This interconnection is achieved through the use of the Generic RDA Service and Protocol standard that defines a service facility that represents a boundary between the local processing of an application and the part concerned with communication. RDA may be used to carry database language commands and data between a client process and a database server to enable an application to read and update data in a remote database. Such commands must be established via a "specialization" of the Generic Service and Protocol standard. A draft-proposed SQL Specialization for RDA has been developed by ISO JTC1/SC21/WG3. Primary committee emphasis has been on the Generic RDA Service and Protocol standard [ISO DP 9579-1] and the RDA Specialization standard [ISO DP 9579-2]. RDA specializations for SQL2 and IRDS are planned by WG3.

The Generic RDA Service and Protocol standard recently completed its second ISO ballot, and the SQL Specialization standard recently completed its first ISO ballot. The results of these ballots will be made public in early 1991. However, due to reputed technical deficiencies, it is doubtful that they will be accepted until some time in 1992. There is increasing vendor interest in the RDA standards among vendors in the United States. Thus, it is likely that the commercial DBMS industry will produce RDA implementations.

B.4.5 POSIX Database Services

A POSIX Open Systems Environment is a set of ISO, regional, and national information technology standards and functional standards profiles that specify interfaces, services, and supporting formats for interoperability and portability of applications, data, and people that are in accord with ISO 9945 (POSIX). Appendix D of this document provides a detailed description of the POSIX environment. The current draft of the Guide to POSIX Open Systems Environments (P1003.0/Draft 6) contains a section on Database Services Requirements that describes how the conventional view of database management is related

to the POSIX system architecture. The document identifies the database services that must exist in a POSIX compliant system: data definition and manipulation services, data access services, data integrity services, and miscellaneous services. Currently, most of these database services can be found in commercially available relational database management systems. It appears that POSIX will be more a consumer of data management standards than a producer. As data management related standards are developed they will be considered for inclusion into the POSIX open systems environment.

B.4.6 Other Data Management Related Efforts

Many other data management related efforts within ISO are at early stages of discussion, standardization or acceptance. This appendix identifies only effort known: Distributed Transaction Processing (DTP) [ISO DIS 10026-1], Open Distributed Processing (ODP) [ODP 1987], OSI Directory [ISO 9594-1], Data Element Standards, and Concepts and Terminology for the Conceptual Schema and the Information Base [TR 9007].

B.5 ISSUES

For at least the next ten years, data management technology is likely to be dominated by three issues: distribution, multiple data models, and security. The clear goal that encompasses all three of these is the ability to support distributed, heterogeneous, multilevel secure databases. There are two main questions: When will the technology be available? How will current standards evolve to integrate the new technologies?

The technology to support distributed, heterogeneous databases is already beginning to emerge, but the security aspects are likely to take a long time. It is not yet clear what roles various data models will play. There is neither a good theoretical specification for which data model should be used for what purpose, nor enough experience for a consensus to have formed on the issue. As a result, the rate of standards development is hampered, and various organizations are going in different directions to build ad hoc solutions to current needs. There is some support for the notion that SQL can be the language that links database systems of different models, but not all experts agree. Consequently, the problem becomes one not only of writing the standards, but also getting vendor concurrence with suitable products. If more than one standard emerges, it could take even longer to see if one becomes dominant or if more than one standards is needed to cover different cases.

APPENDIX C —NETWORK SERVICES

C.1 INTRODUCTION

Network Services allow users, applications, operating systems, and database management systems (DBMS) to execute on distributed, heterogeneous computers and carry out fundamental tasks needed to support distributed computing. Such services include exchange of electronic messages, file transfer and access management, directory services, distributed data management, network management, virtual terminals, transferring and manipulating jobs, security functions, and distributed transaction processing. They involve both local area networks (LAN) and wide area networks (WAN) for intra- and inter-site communication.

Technology is constantly evolving. As new technologies emerge, military CCIS users will either evolve or use devices that will become obsolete, more expensive to maintain and interoperate, and slower than those of opponents who modernize. By using accepted frameworks for defining services and developing interfaces, the Network Services should provide transparent support for distributed processing, promote interoperability while allowing participants freedom to select equipment from different vendors, support portability of software applications, allow incorporation of COTS products, and support integration of software applications originally developed for other purposes into the CCIS. Considerations of security could overshadow some of these goals. For example, security might dictate that distributed processing should not be transparent but involve explicit security procedures. Similarly, procurement practices might establish an advantage to buying equipment from a single vendor, or licensing considerations might indicate the need to discourage use of COTS products or portable software.

The discussion of Network Services extends from a description of the standards and protocols that provide services for the application programs to the agreements needed for transfer of signals over physical media. It does not include details of the physical media. Examples of the requirements include:

- a. Connectivity with component databases and reporting systems [NIS ROC, 25]
- b. Capability for dynamically and rapidly reconfiguring the system during periods of outage or priority operations [JCS Pub 6-03.10, Vol II, Annex B, 6]
- c. Use of interactive decision aids [USCENTCOM C³ Master Plan, Vol. I.]
- d. Support for computer graphics conferencing [JOPES ROC, 106]
- e. Incorporation of full multi-level security to include intelligence information [JOPES ROC, 40]
- f. Congruence of data base structures [NIS ROC, 25]

The Open Systems Interconnection (OSI) Reference Model [ISO 7498] serves as the Network Services architectural model. WAM should adhere to the ISO OSI standards for reasons of economy (commercial products will lower procurement cost), availability (procuring COTS systems will reduce development and implementation time), and use of existing services (DoD will be able to use existing commercial services for some applications).

Services and protocols are described relative to each of the seven OSI layers. The layers are defined in international standards, draft international standards, recommendations of groups, national standards, and other agreements. The objective of the OSI model is to aid exchange of information among computers. However, the protocol definitions themselves contain many optional features and simply agreeing to a protocol will not ensure successful system interconnection. Agreement on the optional features is needed. Those using the model must choose options for the entire network and must determine how each protocol program will provide each service. The agreements, protocol implementation conformance specifications (PICS), contain implementation detail for each protocol selected. Agreements on groups of services and protocols that implement common data communication services are contained in the profiles.

As noted in Section 3, the CCIS will be based on the distributed model of computing, in that a command center has a number of computers connected to a network. The relative proximity of the machines allows the use of a local area network. Within a command center, high speed LANs will be engineered to provide the best overall combination of cost, capacity, and performance. In contrast, the wide area networks connecting widely-separated parts of the CCIS trade the capacity of any computer against its ability to communicate cost effectively over distance. All networked communications will be GOSIP compliant. This

allows easy incorporation of cost-effective GOSIP compliant COTS products into the CCIS. The OSI model has wide support within the European community, Japan, and Korea, as well as in the U. S.

The current WWMCCS architecture consists of approximately forty-three command centers connected by a backbone network. The centers consist of Honeywell computers with associated data storage devices, terminals, and communication processors [DCP 1989, C-2]. The backbone, Defense Data Network (DDN) and AUTODIN, uses leased land lines and satellite channels to exchange packets of data using the ARPA 1822 protocol [BBN 1981]. This was a predecessor to the widely used Transmission Control Protocol (MIL-STD-1778) and underlying Internet Protocol (MIL-STD-1777), called TCP/IP. The backbone links vary in bandwidth (9.6, 50, 56, and 64Kbps). Neither the backbone network nor the LAN comply with the OSI model; thus both must be replaced if the system is to be compliant with GOSIP [FIPS 146].

An important trend in technology is the availability of greater bandwidth in both WANs and LANs. Fiber optic communications provide the system designer the option of having LANs with up to a hundred Megabits per second of throughput capacity and WANs with even greater capacities. There is also movement toward greater integration of functions into a common utility. Within the telecommunications industry, there has been extensive discussion of designing a more intelligent communication system providing a range of services through a single connection. The emerging Integrated Services Digital Network (ISDN) and Broadband ISDN will require the development of new standards, because they offer the potential for multimedia data (voice, video quality pictures, etc.) over the same cable. They separate control from data signals through "out-of-band" signaling. Because newer networks support closer cooperation among dispersed sites, the issue of latency (the elapsed time between the time that an item of information leaves the sender and when it arrives at its destination) is gaining importance. If users perceive an unacceptable delay in response to requests sent to remote sites, they may demand that the remote resources be replaced by local devices.

C.2 THE OSI/GOSIP LAYERED MODEL

The OSI and GOSIP layers shown in Table C-1 , define a model for connecting computers so that they can operate together despite differences in design. The model achieves

Table C-1. Layered Model of Communications

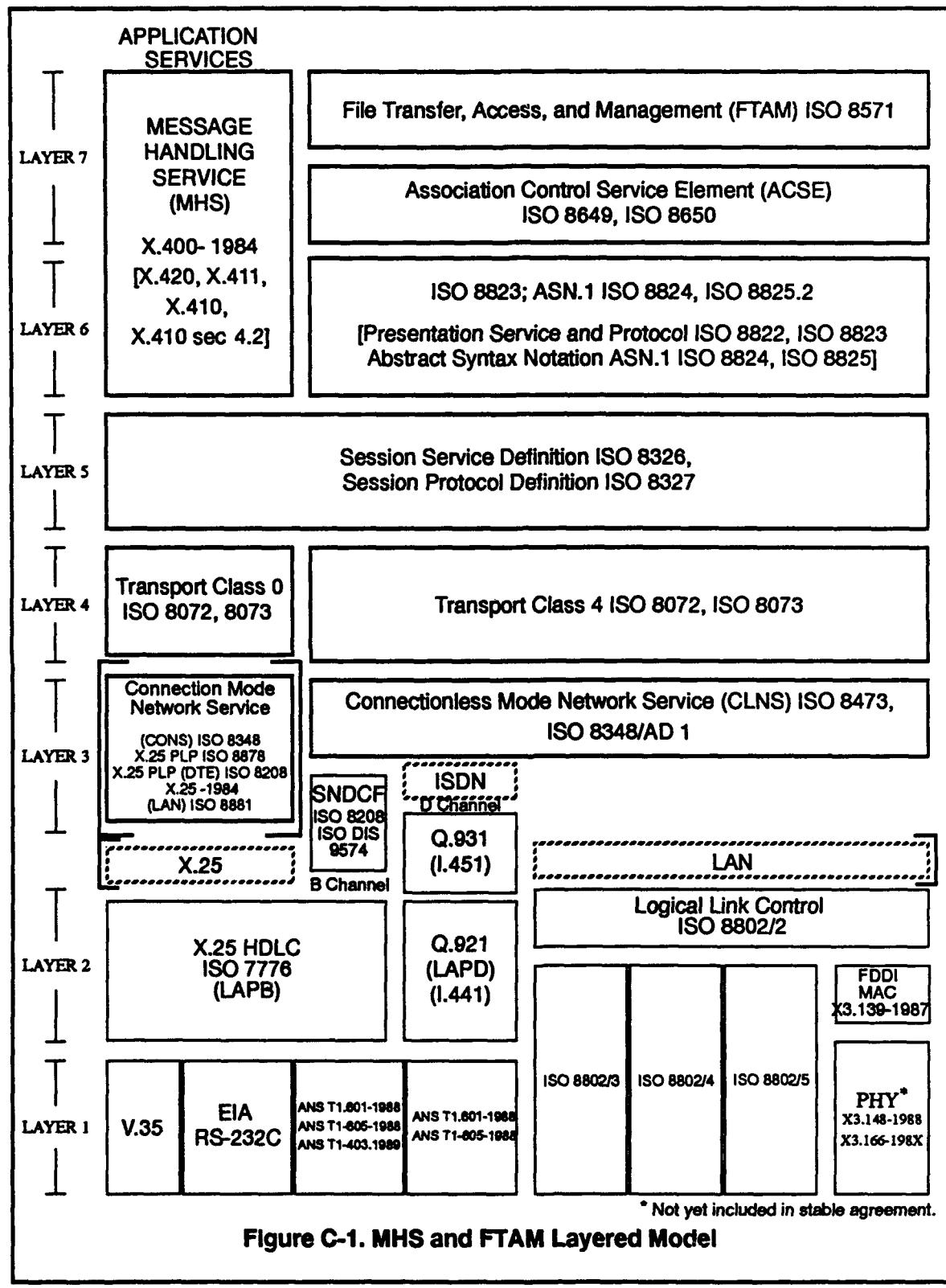
Layer	Name	Function
7	Application Layer	Provides all OSI services to the application processes.
6	Presentation Layer	Establish session, transfer data, negotiation of syntax, transformation of syntax, formatting, and special purpose transformations, session termination.
5	Session Layer	Session connection establishment and release, normal and expedited data exchange, quarantine service, interaction management, session-connection synchronization, session-connection flow control, and exception reporting.
4	Transport Layer	Mapping transport-address onto network-address, multiplexing end-to-end transport connections, establishment and release of transport-connections, end-to-end sequence control on individual connections, end-to-end error detection and recovery, flow control, expedited transport service data unit transfer, and supervisory functions
3	Network Layer	Routing and relay, network connection including multiplexing, segmenting and blocking, error detection and recovery, sequencing, flow control, expedited data transfer, reset, service selection, and Network Layer management.
2	Data Link Layer	Data link connection establishment and release, delimiting and synchronization, sequence control, error detection and recovery, flow control, identification and parameter exchange, control of data circuit interconnection, and Data Link Layer management.
1	Physical Layer	Physical connection activation and deactivation, physical service data unit transmission, and Physical Layer management.

its goals by allowing peer layers (those at the same level) to implement a protocol that accomplishes the function of the layer. They coordinate the protocol by exchanging information contained in coded headers that are appended to the packet of information being passed vertically through the layers. The term for this header information is protocol con-

trol information (PCI). The PCI and SDU (service data unit) together form the protocol data unit (PDU), the packet of information passed down a layer and ultimately over the network. The OSI model can be thought of as a series of envelopes used by the sender to forward information. Each layer receives an envelope (PDU) from the layer above, along with a request for a particular class of service. The lower layer interprets the request and translates it into instructions that are written onto the outside of a new envelope that contains the one received from above. The new envelope is passed down a layer. At the lowest (physical) layer, the set of envelopes are sent to the destination, where each layer reads the instructions on the outer envelope, takes the appropriate action, opens the envelope and, if further action is required, passes the contents to the layer above.

Protocols as currently defined have many options. Unless users agree on the options that they will implement, peers will not be able to interconnect. NIST hosts OIW (OSI Implementor's Workshop) meetings whose goal is agreement on options. The meetings develop consensus on the protocols needed to achieve open system goals. After each quarterly meeting, a text of stable implementation and ongoing agreements is published, becoming the basis for new GOSIP versions. Much of the information below is drawn from OIW reports. Other OIW documents also describe "Ongoing Implementation Agreements for Open Systems Interconnection Protocols: Continuing Agreements," and "Working Implementation Agreements for Open Systems Interconnection Protocols." These contain information expected to be a basis for future agreements, but not yet ready for use in product development or procurement. Figure C-1 shows an example of the layered model applied to two network applications, Message Handling Service (MHS) and File Transfer, Access, and Management (FTAM) [ATTCIS 1989, 93, Boland 1989a]. The figure does not show an application program using these network applications but is intended to illustrate:

- a. Designers implement a service by selecting a profile stack, the set of protocols formed by moving through the figure from top to bottom following the rules:
 - Enter or exit a rectangle only on the top or bottom edges.
 - Within a rectangle horizontal movement is unrestricted.
- b. There is more than one way to implement the OSI model; e.g., below the Session Layer there are two choices for transport service protocol (Transport Classes 0 (TP0) and 4 (TP4)). The first is for a connection mode network and data link services and the second for connectionless mode network service.

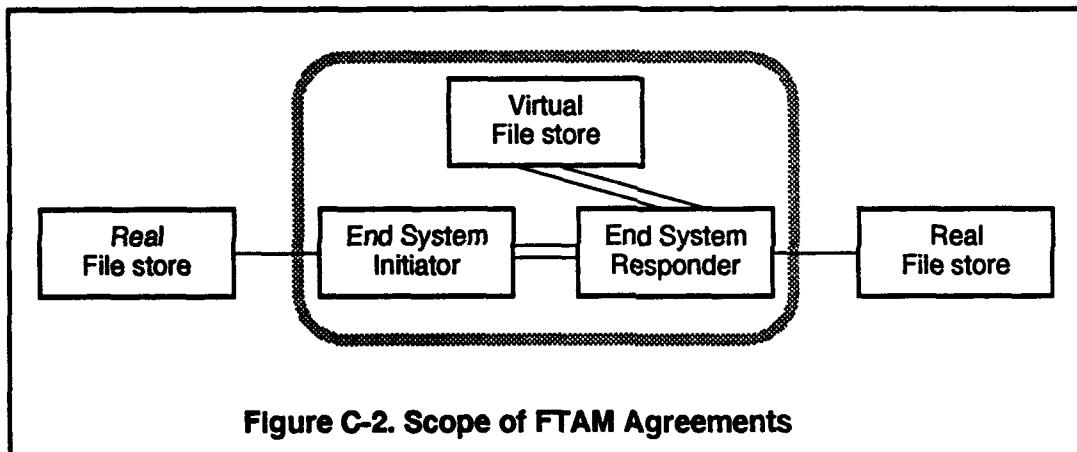


- c. Some older high-level services combine Layers 6 and 7; e.g., the 1984 version of X.400 MHS provides its own Application and Presentation Layer protocols.

C.3 APPLICATION SERVICES

C.3.1 File Transfer, Access, and Management (FTAM)

FTAM has four parts: general concepts, virtual file store, file service, and file protocol [Boland 1989a, 9-1]. It allows an application on one system to establish a connection with an application service element (ASE) on another, providing transfer, access, and management of files in a virtual file store. Figure C-2 shows the relationship of the initiator, responder, virtual file store, and real file stores. (According to ISO 8571, a virtual filestore



is an abstract model for describing files and filestores, and the possible actions on them. A real filestore is an organized collection of files, including their attributes and names, which reside in a real system and to which the virtual file references are mapped. Finally, a virtual file is an unambiguously named collection of structured information having a common set of attributes.) The scope of FTAM is illustrated by the shaded outline in the figure. Only mappings between the initiator, responder, and virtual file system apply. Management of real file stores is not included. FTAM allows access to files on remote systems without regard to differences in formats or syntax. Accessed files appear to be available locally.

C.3.2 Telematic Services

TELETEX is a service that provides communication between terminals used for the preparation, editing, and printing of correspondence. Information is transferred on a memory-to-memory basis [Smith 1981, 7]. Transmission is at a rate of one page in two seconds at 9.6Kbps. CCITT Recommendations F.200, S.60, S.61, S.62, and S.70 define the service, the terminal equipment, character repertoire, control procedures, and supporting transport services for TELETEX. CCITT Recommendation X.430 describes the access protocol for TELETEX terminals.

Within the OIW discussion [Boland 1989b, 8-15 (Table 8.5)] of the X.400-1988 Message Handling System, the Interpersonal Messaging Protocol (IPM) section lists TELETEX as an optional body part type, with a footnote that recommends that classification for this body type be studied further. However, Volume 2, Number 1 of the Working Implementation Agreements states (page 8-19) that there is no intent to develop further agreements for TELETEX access units as part of IPM. Similarly, the April 1989 draft version of GOSIP Version 2.0 does not mention TELETEX among the extensions to X.400. The current NATO ATCCIS transition strategy lists TELETEX as part of profiles T.60 and T.61, that define the character repertoire and coded character sets for international TELETEX services, at the Application and Presentation Layers, and T.62 at the Session Layer [ATCCIS, 23]. Although TELETEX appears to provide a service for that the command center users have expressed interest, it is not now part of the architecture as implementation agreements are not sufficiently mature.

TELEFAX or facsimile has become a widely accepted and available technology. There are many facsimile products available and they could provide an interim solution to some communication problems. However, this service is not part of GOSIP and may not integrate well with other components of a command center. Relevant TELEFAX standards are CCITT Recommendations T.5, T.6, and T.73. The first covers general aspects of Group 4 facsimile apparatus. T.6 addresses the coding schemes and coding control functions for facsimile and Group 4 facsimile apparatus. T.73 defines the document interchange protocol for the Telematic Services. TEXTFAX is a mixed-mode service. It provides a combination of TELETEX and TELEFAX services. This would allow a user to transmit text documents, like TELETEX, as well as facsimile documents, like TELEFAX. It is anticipated that eventually such a service will be included in a command center. This service is not yet well defined.

TELETEX and facsimile encoded documents can be included in MHS documents. There are provisions within the MHS protocols to include TELETEXT and facsimile documents. However, it appears that it will be some time before these services become accepted standards and appear in COTS products.

C.3.3 Message Handling System

Current CCISs provide electronic mail and electronic message facilities. Formal, usually formatted, electronic messages are used to relay detailed command and control information. Informal electronic messages are used to communicate data of a general nature or less detailed command and control information. Defined Network Services standards are sufficient for electronic message transfer within a network, as well as an internetwork store and forward message system.

CCITT Recommendation X.400 has the user interact with the Message Handling System (MHS) through a User Agent (UA) used to complete the message envelope and insert the message contents. The UA interacts with the Message Transfer Agent (MTA) of the MTS to complete the actual delivery of the message. If the destination UA is on the same machine, the MTA may forward the message directly. If the destination UA is on another machine, the MTA may interact with an MTA on the destination machine or on an intermediary machine that is then responsible for forwarding the message to its destination. If the destination UA is unavailable, the messages may be stored and delivered later.

CCITT Recommendation X.400 was specified with the realization that input output devices, "intelligent" terminals, and interconnections with Telematic Services could be required of a MHS. The specification is broad, offering services including access management, a content type indication, delivery time stamp, non-delivery notification, and a submission time stamp. There are also optional specifications for alternative recipients, deferred delivery, delivery notification, grade of delivery services, multi-destination delivery, and return of contents. Some systems may implement a query capability, that allows a user to determine if it is possible to route messages to a particular user. The identification of the originator and destination of the message, as well as the ability to forward messages and create courtesy and blind copies, involve the definition of the contents of the message envelope. The MTA and the MTS use the envelope to determine how and to whom to

deliver the message. The message body is not restricted to simple text but could be a multi-part message with voice, text, graphics, and facsimile components.

The complete description of the MHS, the system model, the available services, formats, and protocols, exists within CCITT Recommendations X.400, X.401, X.408, X.409, X.410, X.411, X.420, and X.430. The MHS model and the IPM and MT service elements are described in CCITT Recommendation X.400. The basic service elements and optional user services are listed in CCITT Recommendation X.401, while CCITT Recommendation X.408 defines the algorithms for type conversion of messages and CCITT Recommendation X.409 presents the notation and encoding technique used in an MHS. CCITT Recommendation X.410 describes how Open Systems Interconnection protocols can be used by MHS applications. The Message Transfer Layer protocols are described in CCITT Recommendation X.411. CCITT Recommendation X.420 specifies IPM service protocols and the use of TELETEX terminals with a MHS is described in CCITT Recommendation X.430.

ISO standards for MHS are called MOTIS (Message-Oriented Text Interchange System) and are defined in ISO 10021. There are some differences between MOTIS and MHS. However MHS is mandated by GOSIP and hence, adopted for the CCIS architecture.

C.3.3.1 Virtual Terminal

Virtual Terminal (VT) is a service specified in ISO 9040, ISO 9041, and is included in GOSIP, beginning with version 2.0. The ISO standards (9040 and 9041) define the protocols and services available as a part of VT. The protocols and services associated with VT can be used to describe a large number of terminals or input/output devices. These devices can range in capability from "dumb" terminals to "intelligent" devices. Character oriented or dumb terminals operating on a character or a line at a time are described by the Virtual Terminal Basic Class or the GOSIP simple system. They include the ASCII character set and its control characters. The interpretation of control characters includes the ability to implement form feed, carriage return, horizontal tab, and back space functions. GOSIP specifies that this mode of service must be able to support the TELNET profile, that specifies a simple delivery control and asynchronous mode operation.

The other category of VT systems specified by GOSIP includes systems capable of displaying and interacting with forms. A system capable of form-handling provides added features such as, cursor movement, screen erase, and field protection. The ISO standard

protocols and services are able to describe terminals or input/output devices that are not a part of GOSIP. There are definitions for two dimensional and three dimensional operations within the VT protocols and services, but these are not a part of the GOSIP profile.

C.3.4 Distributed Transaction Processing

As described in ISO 10026, a transaction is characterized by four properties: atomicity, consistency, isolation, and durability. *Atomicity* means that all or none of the operations of a unit of work are performed. *Consistency* implies that if the operations are performed, then all are performed accurately, correctly, and have validity with respect to the application. The data of the transaction are transformed from one consistent state to another. *Isolation* implies that partial results of a unit of work are part of that unit of work. It also implies that units of work that share writable data are done in sequence. *Durability* means that none of the effects of a completed unit of work are altered by any failure.

A *distributed transaction* is one that spans more than one subsystem, e.g., from one command center to another. Within each processor part of the transaction relates to other parts by a TP-Service User (TPSU). Each part of a transaction is associated with an individual TPSU. The instance of a TPSU that actually performs the functions for a specific occasion is called an TPSU invocation (TPSUI). TPSUIs communicate among themselves in a dialog that includes transfer of data, error notification, synchronization of activities (to reach a mutually agreed processing point), initiation, commitment, and rollback of a transaction, and orderly or abrupt termination of a dialog. The dialog may be controlled by both TPSUIs or by only one of them. Either party can initiate error notification, rollback, or abrupt termination. Dialogs form a tree structure with a root TPSUI maintaining dialog with one or more subordinates, each subordinate of that may in turn have its own subordinates.

A TPSUI involved in a distributed transaction to exchange data will relate to a single application association object (SAO) that will always contain ACSE and commitment, concurrency, and recovery (CCR). The SAO will contain transaction processing application service elements (TPASE). It may also contain user-specific application service elements. All these elements are coordinated by the single association control function (SACF) that provides temporal ordering of the protocol data units over the application association. A TPSUI may have several different SAOs active. Coordination of multiple SAOs is modeled in the multiple association control function (MACF).

C.3.5 Job Transfer and Manipulation

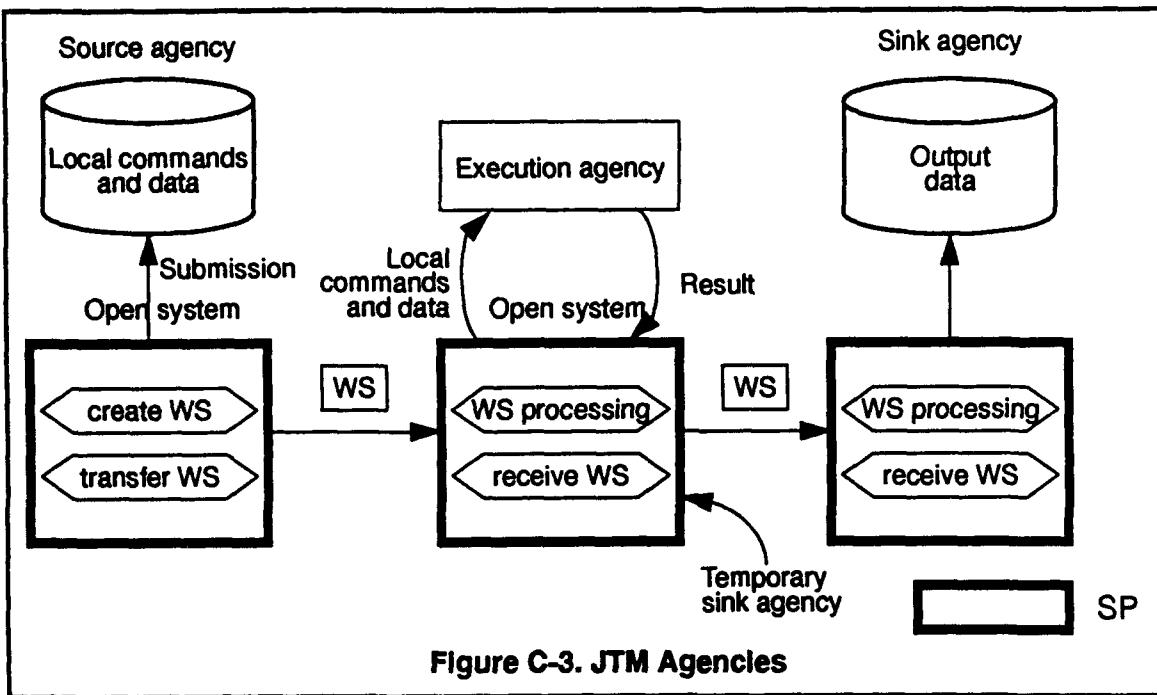
ISO 8831 defines the concepts of and services for Job Transfer and Manipulation (JTM). It requires users to specify the open system where the work is to be done, to know the local functions and its facilities, and to know the control languages used to specify its local work. The standard does not address the standardization of control languages, but within the CCIS architecture, adherence to POSIX can provide such standardization. JTM provides the means to specify the work to be done on an open system, to monitor the execution of previously specified work, and to modify previously specified work. Work specified to be done on one system can initiate work on others.

JTM can be described as independent application entities (AEs) on different open systems. Collectively the AEs form the JTM service provider, serving users called agencies. The interactions between the JTM service provider (SP) and the user agencies are defined in terms of service primitives, indicated in italics by *j-[primitive name]*. The work is performed by the SP using standardized functions that are accessed within the local system environment through the movement of documents among JTM service providers and user agencies. The term *work specification* (WS) refers to documents passed by the SP to the local system or to another open system for storage or for disposal. When an open system completes the work described in the WS, it no longer exists. JTM protocol transfers WS between open systems to accomplish some work. A *proforma* is part of a WS that specifies further work. It is used to form a new WS as part of accomplishing the completion of earlier work. The process of taking data from a proforma and using it to produce a new work specification is called *spawning*. Proforma can contain other proforma nested to any depth. The SP spans WS, using top level proformas and spawning control data in the proformas.

The agencies in the JTM model are the *source*, the *sink*, and the *execution* agency. The source provides documents for inclusion in the WS. The sink receives documents and the execution agency initially acts as a sink for a document but subsequently acts as a source of related documents that are produced as a result of processing a previous document. Figure C-3 shows the relationship of source, sink, and execution agencies.

C.3.5.1 OSI Jobs

The term *OSI job* refers to the complete activity specified by a *j-Initiate* request. A job contains subjobs of various types including document movement, work manipulation,



request manipulation, and transfer control manipulation. The open system upon which the *j-Initiate* occurs is called the *OSI job submission system*. The JTM PDU used to carry details of a work specification is called a Transfer Element. Jobs can be manipulated through modification of a work specification, display, or deletion of reports. The complete job consists of all activities performed either consecutively or in parallel on one or more open systems.

JTM activity consists of a tree of tasks called *subjobs*. These tasks can be performed as a single atomic action (CCR), or each can each be performed as a separate atomic action. When a document is being moved by a document movement subjob, the participating open systems may issue a *j-Dispose* or *j-Give*. The first is issued by a target system to a sink or execution agency to tell them how to deal with a document. An open system collecting a document part does so by issuing a *j-Give*. A system processing either type of JTM primitive may optionally use FTAM to accomplish the actual transfer.

The JTM protocol provides for the detection and reporting of error conditions either by: the open system receiving the WS or open systems that have accepted a WS and are processing it. Error handling can be summarized as one of the following actions:

- a. Accessing a source agency: embed a diagnostic and continue, abnormal termination, and suspend to allow user correction and later attempt
- b. Accessing a sink agency: abnormal termination and suspend to allow user correction and later attempt
- c. Attempting to transfer a WS: abnormal termination and suspend to allow user correction and later attempt

After the initiating agency submits an OSI job, it can wait for the job to complete as an atomic action or monitor its progress while proceeding with other tasks. JTM provides the concept of an OSI job monitor to collect information about the status of work and to make that information available according to instructions. JTM monitors can be classified into *primary* or *secondary*. A primary monitor is determined by the SP at the job submission system. Its data can only be changed by the management of that open system. Secondary monitors are determined by the user through parameters in the initiating service primitive. The user can change this data as the work proceeds.

C.3.6 Remote Database Access

ISO 9579 defines Remote Database Access (RDA) as a service to support distributed database processing. RDA allows programs to read and update data in a remote database. It defines a general model for access to remote data and the operations that can be performed on a remote database. The operations are described as abstract interactions between two users over a communications service. RDA uses ACSE, ROSE, and CCR (if transaction management is needed). RDA has the following specific service elements:

- a. *r-Associate* to create or dialog with a database server
- b. *r-Release* to suspend or terminate a dialog in an orderly manner
- c. *r-Open* to request a data resource
- d. *r-Close* to terminate access to a previously opened resource
- e. *r-ExecuteDBL* to request execution of a database language (DBL) statement in SQL a specified number of times and return the result
- f. *r-DefineDBL* to validate and store a certain database command (a handle is returned to the client)

- g. *r-InvokeDBL* to invoke a previously stored command using the appropriate handle
- h. *r-DropDBL* to remove a stored command using its handle
- i. *r-ReportError* to report error conditions.

RDA describes the services that must communicate to support a remote client with a remote server process, when that process controls a database. The overall model for data management (within which RDA is expected to operate) is the Reference model for Data Management, ISO 10032.

C.3.7 Network Management

Network management provides the services necessary to operate and maintain the communications capability. It provides the capability to monitor communications and identify and correct network problems. ISO is taking a comprehensive look at the entire network management issue and ISO 10164 (Systems Management) and ISO 10165 (Structure of Management Information) will have to be taken into account in the design of any specific CCIS. Network management information can be characterized as belonging to one of five types of data, Fault Management, Accounting Management, Configuration Management, Performance Management, and Security Management information.

Fault Management information includes the ability to create, maintain, and examine error logs. The network management component must accept and act upon error notification, including tracing and identifying network system faults. If needed, the system should be able to perform diagnostic test and correct faults.

Accounting Management information could only be considered in terms of costs. This is important in a commercial environment, but it may also provide information on resources consumed. This is a true when the available network resources are limited and the costs are qualitative (i.e., network available or not). This accounting management also provides the information and services needed to maximize the use of the available network resources.

Configuration Management information identifies the condition and version of all Network Services components. It is used in the routine operation of an open system; that is, it can be used to report on the availability, demand for, and condition of all components. The

associated functions can initialize or close down managed objects and, in general, be used to change or manage the configuration of the open system.

The network management system must be able to monitor, report on, and react to *Performance Management* information collected and analyzed during use of the system or under test situations, including analyses of alternative network configurations or the use of complementary and alternative applications.

The nature of a CCIS places a particular emphasis on security. Network management information should include *Security Management* information and services. This should include the ability to create, delete, and control security services and mechanisms or to distribute security relevant information and report on security related events.

All network management information is collected and maintained in a management information base (MIB), and organized as a hierarchy of objects. Each object belongs to a particular object class with a set of attributes having domains of acceptable values, allowable operations, and defined notifications. Objects can represent actual managed entities or support objects. Examples of managed objects include protocol state machines, connections, and physical resources. The definition of an object or a class of objects may include the identification of the attribute or class attributes, the available operations on the object, behaviors, and possible notifications. A particular object can be completely described by its class definition or possibly by a refinement of the class. The only allowable class refinement is the extension of the object by the addition of new attributes. This definition of Network Management objects does not preclude the definition of composite objects.

The standard set of object manipulation functions include the ability to *get* attribute values, *set* attribute values, *create* objects, *delete* objects, request object *actions* to occur, replace attribute values, add and remove object attributes, and select objects by a filter operation. The CMISE (Common Management Information Service Element) provides these capabilities. The actual interpretation of services and issuing of PDUs (Protocol Data Units) is handled by the CMIPM (Common Management Information Protocol Machine). The specification of the protocol associated with the CMIPM is defined in ISO 9596 Common Management Information Protocol Specification. The associated definition of services is found in ISO 9595. The principle of naming objects, the relationships of objects, and the concept of managed objects are defined in standards ISO 7498-4. The logical structure of network management information is covered in ISO 10165-1.

C.3.8 Directory Services

Directory Services provide common functions that are useful to applications, management processes, and users as they attempt to perform interpersonal and inter-system communications. The directory and associated services allow users to find other users, applications, and systems using imprecise information somewhat like a telephone directory. Specifics on this functionality, its services, and its associated protocols, can be found in the CCITT Recommendation X.400 and the ISO 9594.

Information under the control of the Directory Services component is retained in the Directory Information Base or DIB. This includes public information, application or user specific information, and information unique to the Directory Services system. The DIB is organized in a tree structure (DIT, Directory Information Tree) or hierarchy. Objects are accessed by Directory Services functions on behalf of any user or application. Access to objects or object attributes can be restricted in accordance with applicable security policies. An application that wishes to make use of Directory Services invokes a Directory User Agent (DUA) to add, remove, or modify information in the DIB. It also supports reading, comparing, listing, and searching for information. The DUA uses a Directory Access Protocol to interact with a Directory System Agent (DSA). The DSA may provide the requested service or work with other DSAs to provide the requested service. The DSAs interact through the Directory System Protocol.

Directory Services provides general capabilities that support OSI based applications, management processes, and Network Services. Its provides a mechanism with which aliases and user friendly naming can be implemented. The features can be used on a per user, per system, or global basis.

C.3.9 Interactive Graphics

Network Services supports interactive graphics-based applications. The graphics application will negotiate with Presentation Layer services to obtain communications facilities to support its capacity, robustness, and reliability demands. In peer to peer relationships, these connections could be formed using an appropriate ACSE. Detailed information on graphics appears in the User Interface Services part of Section 4 and in Appendix G.

C.3.10 Teleconferencing

Teleconferencing will soon evolve to a point where a CCIS designer can expect to include this function, providing transmission and reception of video and audio data. The technology is beginning to appear in the commercial marketplace. However, it is far from mature and does not appear to have a definitive set of standards. As standards evolve, they will have to be compared with the ISO OSI standards and GOSIP profile to ensure that they provide the functionality without sacrifice of interconnectivity.

C.4 UPPER LAYER SERVICES

The upper layer services are three in number: Application, Presentation, and Session.

C.4.1 Application Layer Services

The Application Layer provides several types of commonly used services: the Association Control Service Element (ACSE), the Remote Operation Service Element (ROSE), and the Reliable Transfer Service Element (RTSE).

ACSE is an Application Layer service used to create, control, or abort application associations, as defined in ISO 8649 with the protocol defined in ISO 8650. In the OSI Reference Model, an application is composed of a set of Application Elements (AEs) that request applications associated with another AE, through an Application Service Element (ASE). Each AE provides a set of functions or services available within its application context. The application association is a cooperative relationship: two AEs may exchange control information, and subsequently data, through the application association. The establishment and dissociation of an application association can be a cooperative process. An AE can request that an association be established or dissociated and wait for a confirmation. The protocols also support the dissociation of application associations without confirmation.

The ACSE can form an application association that operates in normal or X.410-1984 mode. When operating in normal mode, the ACSE communicates with its peer using presentation-service normal mode, as described in ISO 8822. In this mode, actual transfer of information occurs as defined by the Association Control Protocol Machine (ACPM). The ACPM is a finite state machine representation of the allowable interactions between the ACSE-service user and the presentation-service. The ACPMs of the two AEs exchange

Application Protocol Control Information (APCI) and user information. User information is in the form of ACSE Application Protocol Data Units (APDUs). In X.410-1984 mode, APCI information is not exchanged between peer ACPMs. Rather, the ACPMs respond to ACSE-service users and presentation-service requests to transfer data. This information is transferred without the accompanying control information.

ROSE [ISO 9072] consists of the notation and services provided by an application service element to support interactive applications in a distributed open environment. Interactions between application entities of a distributed application are modeled as one of four remote operations: *bind*, *unbind*, *operation*, and *error*. The standard defines five ROSE services. The *ro-invoke* service enables and invoking application entity to request an operation to be performed by the performing application entity. The *ro-result* service enables the performing entity to return the positive reply of a successfully performed operation. The *ro-error* service enables the performing entity to return the negative of the reply issued by a successfully performed operation. If a user (-u) or provider (-p) detects a problem, *ro-reject-u* and *-p* enable one entity to reject the request or reply of the other.

Abstract syntax notation (ASN) is a tool to define protocols, a method for describing the data transferred between ASCEs by the presentation protocols. The language used to implement the method is called Abstract Syntax Notation One (ASN.1), defined in ISO 8824. It provides a set of rules for describing data types and values. For each application service (e.g., FTAM) the abstract syntax defines data element types (that may or may not be ASN.1 primitives) passed between the Application and Presentation Layers. The respective protocols use ASN.1 rules to define the particular syntax used to specify protocol data units (PDUs). A separate part of the protocol defines the actions that are to take place upon receipt of a particular PDU. For example, the NIST stable agreements for FTAM provide a definition (using ASN.1 rules) for each of the document types supported. These agreements describe the syntax for file access data units (FADU) for various types of documents (files). An example of the semantics would be when using the NBS-ordered flat constraint set, an <insert> action is allowed only at the end of the file. Certain combinations of current FADU (state) at the receiver when an <insert> is received will result in the insertion of a new node, whereas other combinations will result in an error condition.

Because it provides a set of rules defined according to an abstract formal language for describing data types, ASN.1 has been said to resemble the data type definition role of programming languages such as Pascal, Ada, or C. However, ASN.1 does not define the same

types of data as Ada; the resemblance extends only to its existence as a formalism for defining data types. The language has provisions for three types, plus four classes of tags to distinguish the different types. Definitions of types can be grouped into modules that might correspond to a particular PDU.

C.4.2 Presentation Layer Services

The stable agreements of the OSI implementors workshop define a mandatory presentation kernel that supports connection, transfer, and release. The optional services support context management and context restoration. Some of the 1984 ISO application services (MHS and FTAM) did not make use of separate Presentation and Session Layer services, but the 1988 versions of the standards do include these layers. The Presentation Layer also provides a transfer syntax for encoding the PDU it receives from the Application Layer. The transfer syntax describes how the individual will be represented as a stream of binary data. The transfer syntax mentioned in the OSI implementors agreements is either “single ASN.1-type” or “octet-aligned.” The latter option is reserved for EXTERNAL types, encoded as an integral number of octets, and not an instance of an ASN.1 type. ISO 8825 contains the Basic Encoding Rules (BER) for ASN.1. Implementing agreements may provide more restrictive conditions, e.g., the continuing agreements requires any incidence of an ASN.1 INTEGER type used to describe protocol control information (PCI) must be encoded in no more than four octets unless explicit NIST agreement allows an exception.

The basic encoding rules described in ISO 8825 are also referred to as “Basic Encoding of a single ASN.1 type,” or {joint-iso-ccitt asn1 (1) basic-encoding (1)}. The rules provide for four components of encoded data: identifier octets, length octets, contents octets, and optionally end-of-contents octets. Although several encoding schema are allowable under OSI, the only encoding methods mentioned in the NIST Implementors Agreements are the ISO 8825 BER or “octet-aligned.” Connectionless Presentation Layer services are described in ISO 8822 and protocols are described in ISO 9576.

C.4.3 Session Layer Services

The purpose of the Session Layer is to establish the rules for a “dialog” that will take place among the participating entities. The Session Layer also maintains the dialog until the rules for ending it are satisfied [Melendez and Peterson 1987]. It supports the following

functional units: kernel, duplex, expedited data, re-synchronize, exceptions, activity management, half-duplex, minor synchronize, major synchronize, and typed data. Session Layer service definition is contained in ISO 8326 and the protocol definition is contained in ISO 8327. The OIW stable agreements contain options for concatenation so that conforming implementations can accept, abort, or refuse to accept (during negotiation) the following cases: concatenated incoming session protocol data units (SPDU) containing category 0 SPDUs and category 2 SPDUs with a Token Item field or User Data, or if extended concatenation is requested.

Session segmenting and reuse of transport connection are not required and conforming implementations must be able to work without them. If transport expedited service is available, it must be requested and used. Session versions 1 and 2 are recognized. Version 2 allows the use of unlimited user data during connection establishment. Cooperating Session Layers having both version 1 and 2 capability can establish connections with implementations having only one of them. It is possible for conforming versions only having version 1 or version 2 to be unable to establish communications. Connectionless Session Layer services are described in ISO 8326 and protocols are described in ISO 9548.

C.5 TRANSPORT LAYER SERVICES

The standard for Transport Layer services is ISO 8073, defining five classes of procedures that correspond to combinations of different levels of service and types of underlying networks. Service classes are designated class 0 through 4 and network types are designated A through C. Only classes 0, 2, and 4 are included in the NIST stable OSI agreements. Only class 4 is authorized for use over connectionless network layer services (CLNS), but all classes are authorized for use with connection-oriented networks.

Class 0 has been designed for use with type A networks (networks with acceptable residual error rates and rate of signaling errors). It provides the simplest type of transport connection and is compatible with the CCITT Recommendation for TELETEX terminals.

Class 2 provides a way to multiplex several transport connections onto a single type A network. It allows optional use of explicit flow control, depending on the type of terminals involved, the level of traffic, and the importance of expedited data.

Class 4 is designed for use with type C networks (those with unacceptable residual error rate. It provides the characteristics of class 2 plus recovery from network disconnect or reset and detection and recovery from errors that occur as a result of low grade of service, including transport protocol data unit (TPDU) loss, delivery out of sequence, TPDU duplication, and TPDU corruption.

The NIST OIW agreements contain provisions for transport class 4 to support four priority level: low, normal, high, and high reserved. When priorities are used, both the network layer services and end system must support them. Mandatory congestion avoidance is the other service included in the current version of the working agreements not yet included in the stable agreements. The stable agreements for transport class 4 provide for "extended formats" and expedited transfer plus optional use of protection, acknowledgment timer, and quality of Service (QoS). The current version of ISO 8073 does not include functions for monitoring QoS or exchanging status information. The layers above and below Transport, Session and Network, do contain functions for QoS. Procedures for passing QoS information between these layers are not clear. Architectural work in ISO to address QoS issues is progressing very slowly [ATTCIS WP-25].

C.6 LOWER LAYER SERVICES

The Network, Data Link, And Physical Layers are considered lower layer protocols. They are concerned with ensuring the desired quality of service expected by upper layers, sequencing traffic, segmenting, flow control, error detection and correction, and control of physical signaling mechanisms.

C.6.1 X.25 WAN

The CCITT Recommendation X.25 protocols were developed to provide access to public, packet-switched networks. They were intended to provide more efficient use of communications resources by users whose demands were "bursty," i.e., that the peak data transmission rate is much higher than the average transfer rate. The peaks are assumed to occur at random, with unpredictable duration. Such conditions are approximated by several important classes of computer users, such as users of electronic mail, remote, interactive users who communicate with a computer by using a terminal, or a network of computers

processing a mixture of jobs that have varying need for data exchange, synchronization, computation, and control.

The X.25 protocols consist of three levels: packet level, link level, and physical level. They support three types of communication, switched virtual circuits, permanent virtual circuits, and (more recently) connectionless service. Like other standards, X.25 has evolved. The 1980 version has been replaced by the 1984 version, but many installations operate with only the earlier version. The NIST OIW recommends that until the 1984 X.25 becomes available on a widespread basis, upper layer protocols (CCITT Recommendation X.400, MHS) use the 1980 X.25 even though the earlier version has limitations.

C.6.2 Connectionless Network Service

ISO 8473 defines the connectionless network service (CLNS) protocol that can be used with X.25 as well as with ISO 8802/2 LAN and ISDN services. For all underlying systems, the OIW has agreed not to use the inactive subset of ISO 8473. Non-segmenting will not be used and PDUs will be generated with a segmentation part. A PDU that does not contain such a part will be correctly received and processed. Other functions provided by implementations conforming to the OIW agreements will have the following mandatory functions: a lifetime parameter (set at least 1.5 seconds), a reassembly time, and, when error reporting PDUs are supported, the contents of the source address field of the PDU generating an error. Optional functions include the following:

- a. The security parameters may be defined by bilateral agreements.
- b. Partial and complete source routing will not be required because of a defect in the partial source routing option.
- c. Partial record of route will be supported by intermediate systems.
- d. QoS will be followed as described in ISO 8473.
- e. Notification of congestion will be provided.

The specific provisions for providing CLNS over X.25 networks are defined in ISO 8348/Addendum 1 and ISO 8473. (See also: ISO 9068.) The necessary subnetwork dependent convergence function (SNDCF) will be as defined in ISO 8473. SNDCF [Hemrick 1982] is used to enhance the functionality of subnetworks that might not otherwise provide full X.25 services. Because it is subnetwork dependent, the particular function is dependent

on the underlying subnetwork. The default throughput class should be used if available. The X.25 packet level protocol (ISO 8208) will be used.

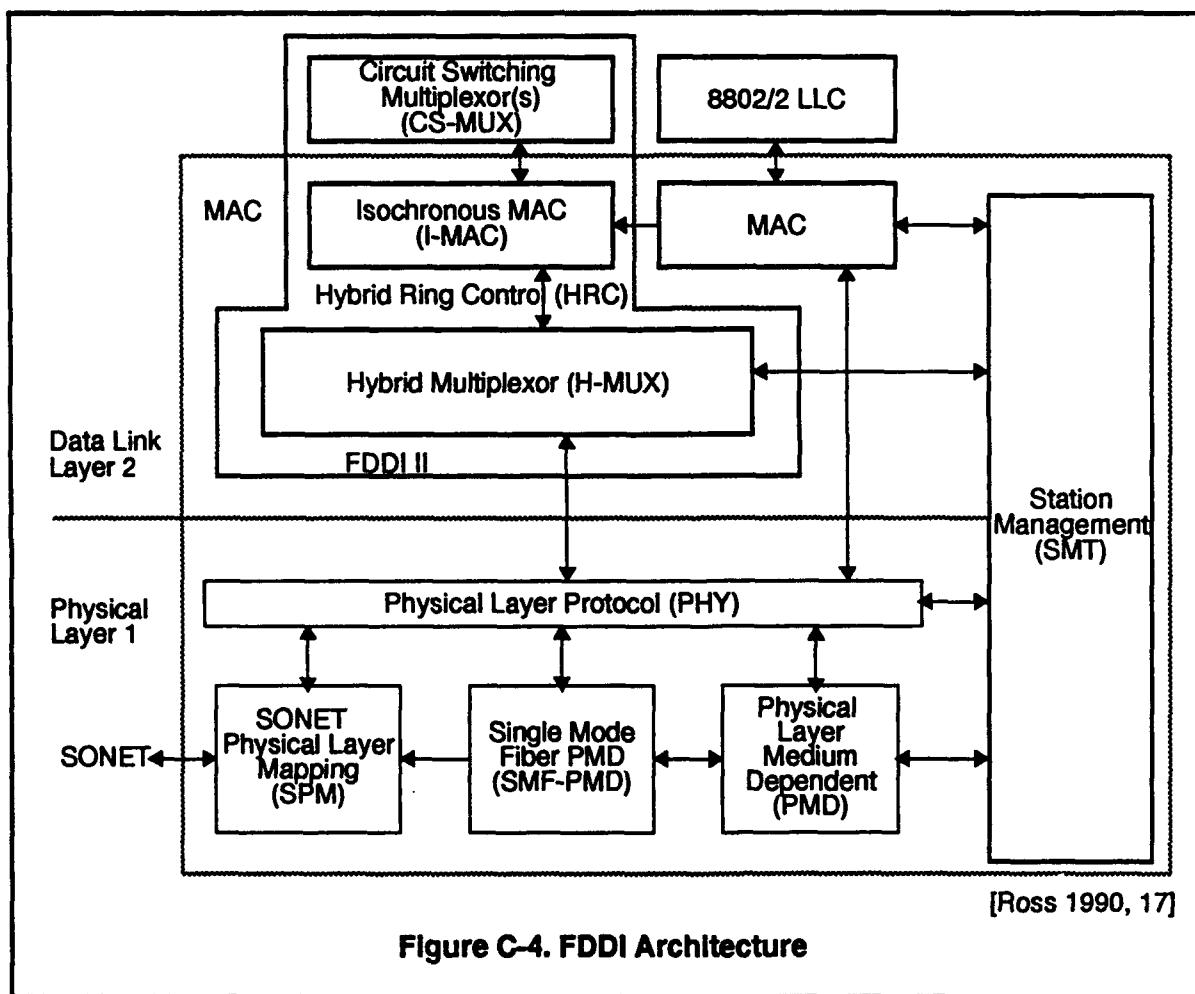
C.6.3 LAN Link Control

ISO 8802/2 defines logical link control (LLC) procedures for several types of conforming local area networks. ISO 8473 and ISO 8348 Addendum 1 define the CLNSs used for LANs. The routing is as described in ISO 9542. Additionally, the stable implementors agreements recommend use of a management mechanism capable of adding and deleting entries in the Routing Information Base (RIB). Additional OIW agreements [Boland 1990, 3-9 - 3-23] regarding ISO 9542 include the following:

- a. Support is provided for any valid network service access point (NSAP) address.
- b. Support is provided for both configuration information and route redirection information, but not for subsets thereof.
- c. Configurability is provided for all timer values.
- d. Configurability is provided for checksums (but not recommended for use in originating protocol data units).
- e. QoS, security, and priority parameters are not to be used for routing
- f. A mechanism to disable configuration notification is needed for broadcast networks.
- g. The same link service access point (LSAP) as ISO 8473 is used for LANs.
- h. The encoding of network service access point (NSAP) address follows the syntax of the data link being used (e.g. 48 bit MAC address on LANs).
- i. The multicast addresses for “all intermediate systems on the network,” and “all end systems on the network” have default encodings.
- j. The error report flag shall be set to zero for network protocol data units (NPDUs) sent as a result of invoking the QUERY Configuration Function.

C.6.4 Fiber Distributed Digital Interface

The Fiber Distributed Digital Interface (FDDI) is a 100Mbps LAN based on a fiber optic token ring protocol [Ross 1990, 16]. An FDDI standard is still under development, but its inclusion in the OIW agreements provides an indication of support among vendors. The protocols that define the FDDI standard consist of a basic set of four protocols and an additional three protocols that are being developed as an extension. Figure C-4 shows the architecture of FDDI [Ross 1990, 17].



Basic FDDI protocols are found in the lower two layers of the OSI model. FDDI conforms to the ISO 802.2 logical link control and the basic protocols consist of Media Access Control (MAC) in Layer 2 (the Data Link Layer) and two protocols in Layer 1 (the Physical Layer). The Layer 1 protocols are called Physical Layer Protocol (PHY) and Phys-

ical Layer Medium Dependent (PMD). A final protocol, the Station Management (SMT), spans Layers 1 and 2. The optional protocols are also found in Layers 1 and 2. In Layer 2, the Hybrid Ring Control (HRC) consists of an Isochronous MAC (I-MAC) and a Hybrid Multiplexer (H-MUX). HRC is not controlled by LLC, but by a Circuit Switching Multiplexer (CS-MUX). There are two additional protocols at the Physical Layer: Single Mode Fiber PMD (SMF-PMD) that uses a laser source and allows fiber links up to 60 km, and SONET PMD (SPM) that provides transport of FDDI over common carrier synchronous optical network (SONET).

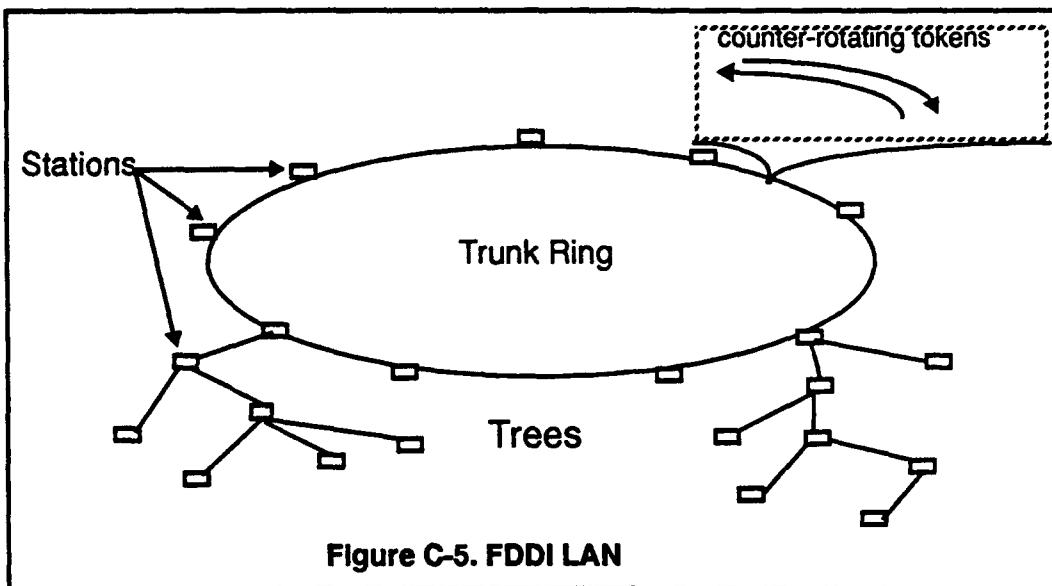
The physical medium used by FDDI consists of dual fiber connections configured as concentric rings. A separate token is passed on each ring as a means of controlling access to the communication resources. A station can transmit data only when it holds a token. Stations can connect either to a single trunk or to both trunks. Stations that connect to both links can have either a single Media Access Control (MAC) or dual MAC. As a result, a station can be one of three types, Dual Attachment Station (DAS) single MAC (DAS/SM), DAS dual MAC (DAS/DM), or Single Attachment Station (SAS). The DAS stations provide a mechanism for bypassing failed stations. In the event that a node or link should fail, the adjacent stations will detect the failure and automatically wrap the two broken rings into a single continuous link.

Although FDDI is based on token passing rings, the basic topology is a dual, counter-rotating (the tokens pass in opposite directions) ring of trees. Trees are formed by master ports that attach to the trunk ring, either to a single ring or to the dual rings. Master ports in turn provide a mechanism for slave stations to attach to the ring. Master ports act as concentrators to form trees that extend from the basic ring. Figure C-5 shows the basic trunk ring (counter-rotating tokens) with attached trees.

FDDI PMD specification calls for stations to repeat all valid packets under all signal conditions with a bit error rate of not more than 2.5×10^{-10} . The minimum average power shall be -29 dBm. Link errors on FDDI links are estimated by a Link Error Monitor that excludes whenever a threshold is exceeded. The threshold can be set as high as 10^{-4} , but the nominal value is 10^{-7} . The threshold is specified by the SMT. The link error rate threshold is necessary to ensure that MAC layer operates correctly.

The physical cable for FDDI is either based on 62.5/125 micron multi-mode fiber (PMD) or 8/125 micron single mode fiber (SMF-PMD). The multi-mode is used for distances up to 2 km and the single mode is used up to 40-60 km. Optional

multi-mode fiber sizes are permitted and the proposed standard (ANSI X3.166-199x) contains data on different loss factors for other common fiber sizes.



There is an optional provision in FDDI that allows FDDI service to be transported over common carriers that comply with the North American standard synchronous optical network (SONET). SONET is the architecture used by common carriers in North America to implement the Synchronous Data Hierarchy (SDH). By providing synchronized communication and a virtual container concept, SDH simplifies the design of multiplexors and other network interface devices. It carries traffic in synchronized modes at rates that are integral multiples of 155.52Mbps. The SONET architecture will be compatible with broadband ISDN.

FDDI II introduces a circuit switched service, called isochronous service, to the packet switched service provided by FDDI. Circuit switching can be tailored at various bandwidths from 8, 16, 32, and 64Kbps plus multiples of 64Kbps up to 6.144Mbps, called a wideband channel (WBC). Networks that support isochronous service maintain a synchronized, periodic structure of preamble, cycle header, sixteen WBC, and twelve packet data groups with a cycle time of 125 microseconds. The HRC standard introduces H-MUX and I-MAC, in the Data Link Layer to maintain the cycle structure and control access to the WBC. Unused WBC bandwidth is made available for packet service.

The group that developed the FDDI standard has proposed a new family of standards, the FDDI Follow-On LAN (FFOL) [X3T9/90, X3T9.5/90, FFOL-019, FDDI Follow-On LAN (FFOL) 24 August 1990]. The target date for completion of the basic FFOL standards is December 1995. The estimated life of the FFOL family of standards is 10 to 15 years. General requirements for FFOL include the a backbone for multiple FDDI networks, efficient interconnections to wide area networks, support for a wide variety of integrated services such as data, graphics, video, and audio, an initial data rate matched to SDH and less than 1.25 Gbps, and the ability to use existing FDDI cable plant, where feasible.

C.6.5 ISDN

ISDN (Integrated Services Digital Network) is the result of the current evolution of the networks and services available from the various common carriers. The original telephone or telegraph networks were based on analog equipment. In recent years, analog equipment has been replaced with digital equipment. This has led to the replacement of the analog network with an Integrated Digital Network (IDN) that incorporates the latest in digital switching and transmission. The extension of the IDN to provide additional user services has resulted in the Integrated Services Digital Network (ISDN) that makes an all digital interface available to the network subscriber. Digital switching provides a fast connect and call setup time for voice and data communications. An ISDN node can connect to packet based and circuit based switched or nonswitched networks. The ISDN provides a digital interface that allows the user to connect digital devices directly to the network. The ISDN node interface is referred to as the CRF (Connection Related Functions). User terminal equipment is typically connected to an ISDN node by a NT1 (Network Termination 1) or NT2 (Network Termination 2). Terminal equipment provides basic protocol handling and interface functions including digital telephones, data terminal equipment, and integrated workstations. Terminal Equipment that features an interface that complies the ISDN user-network interface is designated TE1 (Terminal Equipment Type 1). Terminal Equipment Type 2 (TE2) is Terminal Equipment that adheres to an interface recommendation other than ISDN.

Network Termination equipment can be of type 1 (NT1) or type 2 (NT2). Network Termination 1 equipment provides the proper physical and electromagnetic termination of the network. This includes line transmission termination, timing, power transfer, and interface termination. A NT2 provides additional features that can include switching, concentration,

and multiplexing. Examples of NT2 equipment are private branch exchange (PBX), automatic branch exchange (PABX), a local area network (LAN), or a terminal controller. The exact configuration and connections to an ISDN node depend on the capabilities and features of the equipment. Facsimile, TELETEX, and slow scan television equipment could function as terminal and NT1 equipment. It could be directly connected to the ISDN node. Less capable terminal equipment may require connection to an NT1 or NT2 connected to the ISDN node. Terminal Equipment may be intentionally connected to an NT2 rather than an NT1 to take advantage of the multiplexing and concentrating capabilities.

An ISDN node's capacity is determined by the number and kinds of communications channels available to the node. A basic ISDN node has two B channels and one D channel. B channels, 64Kbps, are normally used for the transmission and reception of data. D channels, 16Kbps, are normally used for signalling or control functions. Higher bandwidth H channels may be available with designated bandwidths of 384Kbps, 1536Kbps, or 1920Kbps. The protocols, services, and interfaces to an ISDN network are defined in the CCITT I series recommendations. For example, the CCITT Recommendation I.430 specifies bit, octet, and channel synchronization, as well as, the D channel and access control. CCITT Recommendation I.440 describes the data link logical connection by the D channel, including the ability to transfer a packet by the D channel. The protocol used on the D channel is LAP D, based on HDLC. The CCITT Recommendation also indicates how to establish and clear a call through a circuit switched or packet switched network. CCITT Recommendations I.440 and I.441 jointly provide ISDN service and protocol definitions and specifications, describing how to establish a network connection in a circuit or packet switched network, protocols for transferring a data packet or datagram over a connectionless network, and how to perform the same task using a virtual circuit in a connection-oriented network. The physical connection between terminal equipment and an NTE is covered in CCITT Recommendation I.430. The network connection between the terminal equipment and an NTE is specified in CCITT Recommendation I.440. CCITT Recommendation I.450 indicates how a connection across an ISDN network between two terminals is accomplished.

There is a correspondence between the CCITT ISDN Recommendations and the ISO Reference Model. The physical connection between an ISDN network and a NT1 corresponds to ISO Layer 1. The connections that establish a network connection from an ISDN

network to a NT2 corresponds to ISO Layers 1-3. The teleservices that could be available from an ISDN network correspond to Application Layer services or ISO Layers 1-7.

C.6.6 BISDN

BISDN (or B-ISDN) refers to Broadband ISDN. BISDN overlaps ISDN, but can include a switched or nonswitched network connection that operates in a circuit or packet mode network. It can offer different forms of applications and communications capabilities for distribution-oriented and interactive services such as conversational, messaging, retrieval services, and distribution services with and without individual presentation control. Conversational services provide bidirectional (although unidirectional could be included) dialogue communicationsz that could include video surveillance, videotelephony, video teleconference, videotex, and high speed data communication. Message services extend mail functions to include moving pictures, high resolution images, and audio. These services allow a user to create, edit, process, convert, store, and forward messages. This service allows end users to communicate with each other. Retrieval services allow an end user to retrieve information from a central location or an archival site on demand.

BISDN services can require substantial network capacity. In addition to the ISDN B and H channels, BISDN must support the H21 channel at 32Mbps, the H22 channel with capacity up to 45Mbps, and the H4 channel with a rate as high as 138.24Mbps. The user interface standard rates will be approximately 150Mbps and 600Mbps.

C.6.7 LAN

Local Area Network (LAN) technology was designed to support communications between computing devices using a physical communication channel of moderate data rate within a moderately sized geographic area. The LAN has found wide acceptance in the marketplace. The ISO 8802 series of standards define LAN services, protocols, and functions. ISO 8802 focuses on the definition of a multi-access LAN within which any node can determine if another node is transmitting or, if appropriate, start transmitting. ISO 8802 is applicable to baseband systems where the data rate will be 1-20Mbps, and the total network length less than 2 km.

A LAN for a CCIS may have to operate in one of two very different environments. The first has light network loads, low throughput requirements, and small delays. The other has demanding requirements: heavy network loads, potentially long delays, and a throughput requirement for gradual or controlled degradation. This has resulted in a group of ISO standards. ISO 8802/1 corresponds to the OSI Physical Layer, specifying cable or media for communications and aspects of the media access unit. ISO 8802/2 corresponds to the OSI Data Link Layer, defining aspects of the media access unit and logical link control. Three ISO standards correspond to the Layers 1 and 2: ISO 8802/3, 8802/4, and 8802/5. They define how information is transmitted through the network and are distinguished by the protocols used to control and transmit data packets. An ISO 8802/3 network is based on CSMA/CD (Carrier Sense Multiple Access/Collision Detect), an ISO 8802/4 network is based on token bus concepts, and an ISO 8802/5 network uses a token ring. These protocols differ in their complexity and exhibit different tolerances to loading. For example, an ISO 8802/3 network features a low overhead and works efficiently at a 30% network load. If loading goes beyond 70%, it degrades rapidly as collisions increase. The token based protocols perform better under heavy loading, since, traffic that would otherwise result in collisions is queued at the node.

ISO 8802/3, Ethernet, is unslotted persistent CSMA/CD with binary exponential back-off. In this network, all nodes have equal access to the network. A node wishing to transmit checks the network and if no activity is detected, begins transmission. The node monitors the network while transmitting and if it detects another node's transmission, it retransmits its data packet later. A CSMA/CD network works if nodes are able rapidly to detect an idle network. The network becomes less efficient as propagation and detection delays increase. It also becomes less efficient as channel or data rate increases and data packet size decreases. The maximum throughput of Ethernet depends on the arrangement of the nodes with respect to the cable. Ethernet is nominally rated at 10Mbps.

The ISO 8802/4 (Token Bus) and ISO 8802/5 (Token Ring) networks are more complex than CSMA/CD. These networks require the management of tokens, a process that can be complex. However, token-based systems do not suffer the rapid degradation of service under heavy network load, as does CSMA/CD networks. Depending on the configuration, a token-based network may have a point of failure at each node. Each node could be expected to receive, process, and send tokens. If any node fails to do so, the entire network becomes unusable.

In a token bus network, a network access token is passed from node to node in a round robin. A node that receives this token is granted access to the network to transmit data packets. The addition or deletion of nodes from the token bus network, is accomplished by the use of a special "add node" tokens. The occurrence of this token allows the actions necessary for reconfiguration of the network to occur. A token ring network passes a series of tokens around a logical ring. These tokens can be busy tokens or free tokens. Busy tokens have data within them and are forwarded to the next node in the network. If a node has data to transmit, it selects a free token and uses it to transmit the data. A logical ring configuration is subject to failure at any node. The ISO 8802/5 addresses this concern by specifying the use of a central hub. Token ring networks have an apparent capacity limit of 4Mbps.

C.7 NETWORKING ISSUES

The OSI Reference Model and its associated standards are based on computing and communications abstractions. They have not been fully tested by actual implementation. It is possible that flaws in the model will be uncovered during future implementation and operations and standard addenda issued. Ad hoc standards and DoD standards such as TCP/IP are the result of practical experience, primarily with ARPANET and in the open marketplace. Such standards may include layering concepts of the OSI Reference Model but are not necessarily compatible with the model. The design of a communication system will need to address particular user requirements. Military communications systems have mobile hosts, robustness, and real time communications needs, as well as unique requirements for security and performance. While all issues discussed below are not peculiar to the CCIS environment, they provide potentially useful background information to those planning a specific system implementation.

C.7.1 Security

The security requirements of CCIS communications are stringent. Disclosure of information or denial of service could adversely affect national security. Among the more important current network security standards efforts are the Secure Data Network System (SDNS) and the Standard for Interoperable LAN Security (SILS), IEEE 802.10. SDNS identifies encryption-based features that will increase the trustworthiness of the OSI and GOSIP services and protocols. These services and protocols ensure confidentiality, peer

authentication, access control, and data integrity. SDNS applies to the Application, Transport, and Network Layers. Message Security Protocol provides security features to Message Handling Systems within the Application Layer. Security Protocol 4 (SP4) provides security features within the Transport Layer and Security Protocol 3 (SP3) provides features within the Network Layer. SILS provides security features within the Data Link Layer for local area networks.

C.7.2 Performance

A CCIS will depend on the underlying communications system to deliver messages and data to geographically dispersed sites in a reliable and timely manner, depending on the capabilities and performance aspects of the network media and ability of Network Services protocols to use those capabilities. The selection of network media is not based solely on capacity and performance. Cost, reliability, and security are important. For example, fiber optics provide capacity and performance. However, cellular or radio links are appropriate for mobile sites and meteor burst communications offer "stealth" communications with reduced data rates. Table C-2 identifies existing and potential bandwidths based on wire, line of sight (LOS), and beyond line of sight (BLOS) technologies. The material is derived from references IEEE 1990, Yavu 1990, Elisle 1991, and Steele 1989.

Table C-2. Communications Media Capabilities

Link	Capacity (Mbps)	Type
HF	0.002	BLOS
Meteor Scatter	0.002	BLOS
Meteor Burst	0.002	BLOS
Cellular Radio	0.2	LOS (Actual)
Copper	0.8	Wire
Cellular Radio	2.0	LOS (Target)
Troposcatter	10	BLOS
SATCOM	400	BLOS
Optical	2,500	Cable (Actual)
Optical	100,000	Cable (Theoretical)

The protocols and layers of protocols that provide access to the communications media and allow systems to interconnect in the open systems environment also impact the performance of the communications networks. For example, a radio link described as supporting communications at 400Mbps will supply 312Mbps of bandwidth with the SDH protocol. Reduced bandwidth has not yet been addressed in OSI or GOSIP environments with their additional layers of protocols. Implementations based on these standards are relatively new and the standards are still evolving. One study indicates that FTAM performance between workstations connected by a CSMA/CD LAN is comparable to FTP performance [Pennell 1990]. Currently, due to protocol limitations, CSMA/CD (ISO 8802/3) LANs do not provide efficient file transfer service; the protocols give only about 5% of the available 10Mbps bandwidth. This has resulted in research focused on end-to-end performance and the introduction of experimental protocols. Some, e.g. eXpress Transfer Protocol (XTP) and High Performance Parallel Interface (HIPPI), are more efficient.

One recent study [Lidinsky 1990, 28-33] indicates that file transfer can be supported by throughput ranging from 10Kbps to 1Mbps for file sizes ranging from 10^5 to 10^7 bits. Current upper level file transfer protocols available for workstations connected by a CSMA/CD LAN provide throughput in the range of 1.6 to 3.2Mbps. It is likely that future CCIS applications will require the transfer of very large files, perhaps exceeding 10^7 bits. The CICNet DS-3 Working Group [CICNet 1991] reports that applications such as remote observation, realistic visualization, multimedia conferencing, document imaging, and internetwork video will require possibly gigabits of bandwidth. The bandwidth required to support communications for future CCISs may be in the gigabit range. For example, uncompressed high definition television (HDTV) requires 120Mbps, while standard television requires 30Mbps. High-speed communications links and data compression technology are needed to support these applications. Compression can be used to reduce the bandwidth requirements of a video conference transmission to 2Mbps.

APPENDIX D —OPERATING SYSTEMS SERVICES

D.1 INTRODUCTION

This appendix describes operating system requirements in terms of the services and features needed for a CCIS. The primary thrust is a cross-comparison of operating system services and CCIS requirements. The purpose is to identify the means and degree of support provided by each operating system service for implementing a generic CCIS. Our formation of generic CCIS requirements will be aided by considering a specific implementation, the WWMCCS ADP Modernization (WAM) program.

Special consideration is given to the IEEE Portable Operating System for Computer Environments (POSIX) standards since this family of standards provides the basic system services and is a likely choice for use in defining a CCIS system interface. POSIX is an effort to standardize common interfaces to computer system services. The "X" in POSIX denotes the UNIX heritage of POSIX. Further details on POSIX can be found later in this appendix. POSIX is an important area for U.S. standardization activity and has been recommended for the WAM program.

The end result of this comparison is a measure of POSIX operating system support for the WAM program requirements. This includes identification of the POSIX services that are well matched to the WAM requirements, as well as the deficient areas of POSIX. We expect this analysis would hold to a large degree for a generic CCIS implementation because the WAM program requirements are fairly rigorous.

D.1.1 Scope

Part of this appendix provides a broad discussion of operating system services. Both general and specific operating system services are discussed. We will use the framework provided by this discussion to identify the operating systems services that are necessary to support CCIS requirements.

In a large effort like WAM, it is common for some of the requirements to conflict. Indications of any conflicts will become apparent in the cross-comparison of WAM requirements and operating system services. This appendix does not attempt to resolve the conflict; rather, it notes their existence and examines methods for reducing the impact of such conflicts.

D.1.2 Background

The following sections provide general information on open systems and POSIX, an operating system's view of architectures, and a list of requirements to be addressed within the architecture.

D.1.2.1 Open Systems and POSIX

POSIX represents a key open system standardization effort in the area of operating systems. In the WAM Decision Coordinating Paper (DCP) [WAM DCP, Appendix R], it is recommended that the future architecture of the WWMCCS ADP System adhere to POSIX standards for operating system interfaces. The term POSIX is used to refer to the standards being developed by IEEE¹ Project 1003, which is sponsored by the Technical Committee on Operating Systems of the IEEE Computer Society. The term POSIX is also used to refer to 1003 itself, as well as to the collection of working groups that exist under 1003. In addition, it is used as an umbrella term to encompass not only 1003, but also some closely related IEEE standards projects (e.g., 1201 on windows and 1238 on the API for the File Transfer, Access, and Management (FTAM) protocol).

IEEE Project 1003 consists of a family of working groups. The 1003 working groups are defining interface standards based on UNIX. All 1003 standards are intended to facilitate application portability at the source code level. While UNIX has become the operating system of choice on a large number of widely varying hardware platforms, different interfaces exist for each of several versions. The 1003 working groups are chartered to remedy

1. IEEE rules for naming and numbering use "P" when a standard has an approved Project Authorization Request (PAR), a "T" when a standard is approved for Trial-Use only, an "X" when PAR has not yet been submitted and/or approved, and no prefix is given when the standard is approved nor is it given when referring to the working group. For example, 1003.0 is used when the working group is discussed and P1003.0 is used when the draft standard is discussed.

this situation by defining a standard operating system interface and environment based on UNIX.

The first 1003 working group, 1003.1, has produced a standard now known as IEEE Std 1003.1-1988. IEEE Std 1003.1-1988 defines the interfaces to system services, including process management, signals, time services, file management, pipes, file I/O, and terminal device management. IEEE Std 1003.1-1988, in the UNIX tradition, is oriented toward the interactive multi-user application domain. Using IEEE Std 1003.1-1988 as a baseline, other 1003 working groups are extending application portability to additional application domains.

The set of working groups that currently make up the POSIX family are described in terms of eight categories of standardization activity: guidance, system services, utilities, language bindings, distributed services, windowing, conformance, and profiles [DEC 1990].

The 1003.0 working group provides *guidance* for the coordination of the other individual working groups. This group also attempts to identify gaps in POSIX standards structure and can create new working groups to address those areas. When the standard P1003.0 is finalized, it will provide an overview document describing the scope and content of each standard in the POSIX family, as well as cross-references to related standards.

Working groups in the *system services* area include those for System Services (1003.1), Real-Time Extensions (1003.4), and Security (1003.6). The System Services working group has produced a standard (IEEE Std 1003.1-1988) which defines the interface primarily for low-level routines, but includes some library interfaces. Other working groups addressing application domains use this standard as a baseline. The standard is oriented toward an interactive, multiuser application domain and to centralized computer architectures.

The 1003.4 working group will address issues which are of particular concern to real-time application developers, who must be able to accurately control/predict response times. The initial focus of this working group is on defining application interfaces to the functional areas which impact resource management.

The 1003.6 working group will define interfaces to security services and mechanisms. Its basis for consideration of security issues is the Trusted Computer Security Evaluation Criteria (DoD 5200.28-STD), commonly known as the Orange Book.

Working groups in the *utilities* area include those for Shell and Utilities (1003.2), System Administration (1003.7), and Supercomputing Batch Environment Extension (1003.15). The 1003.2 working group will define a standard programmatic interface to common UNIX utilities (e.g., searching, sorting, pattern matching, etc.), including the command interpreter (shell). The 1003.7 working group will address common administrative functions such as backup, recovery, system start-up and shutdown. The 1003.15 working group will define facilities that provide a network queueing and batch system in a POSIX environment.

Working groups in the *language bindings* area are attempting to remove the C language dependencies from the POSIX standards, motivated partially by the desire to carry POSIX into the international standardization arena. The current plan is to supplement the language-independent standards base with interface definitions for specific languages. Currently, working groups for Ada Bindings (1003.5) and FORTRAN Bindings (1003.9) have been established.

Working groups in the *distributed services* area are trying to define Application Program Interfaces (APIs). Included in this effort are (1) Transparent File Access (1003.8), distributed file systems; (2) Protocol Independent Interfaces (1003.12), a network independent data transport capability; (3) Directory Services (1003.17), working on a standard based on CCITT Recommendation X.500; (4) X.400 Mail Services (1224), based on CCITT Recommendation X.400; (5) File Transfer, Access and Management (1238), based on ISO 8571.

Working groups (1-4) under IEEE Project 1201 are defining a standard *windowing* interface based on X Windows. This interface is being designed to work with any operating system and is not dependent on POSIX.

The Test Methods (1003.3) working group is chartered to develop test methods for measuring *conformance* to POSIX. The P1003.3 standard will define a uniform way of testing systems for conformance to the 1003.1 (System Services); it will specify generic test methods which define how to write assertions and test methods for any POSIX standard. Additional documents will address each standard separately (e.g., P1003.3.1 for System Services and P1003.3.2 for Shell and Utilities).

Working groups in the *profiles* area are trying to define Application Environment Profiles (AEPs). An AEP is a collection of interface standards tailored to a particular application domain. Applications conforming to a particular AEP would be portable across

systems which implement that AEP. The AEPs currently being defined include (1) Supercomputing AEP (P1003.10); (2) Transaction Processing AEP (P1003.11); (3) Real-Time Processing AEP (P1003.13); (4) Traditional Interactive Multiuser System AEP (P1003.18 TIMS); and (5) Multi-Processing Support AEP (P1003.14).

D.1.2.2 Generic CCIS Architecture

In the ensuing discussion, a generic CCIS architecture is examined to develop a basis for further discussion. This generic architecture is viewed as a series of layers. The layers range from the simple case of a stand-alone processor to the heterogeneous loosely coupled network of parallel and sequential processors. All four types of systems are possible components of the generic architecture. The four layers are represented in the matrix shown in Table D-1.

Table D-1. Layers of Operating System Architectures

	Single OS	Multi OS
Multiple Systems layer	Network Mach	Network Apple Network UNIX
Single System layer	VAX UNIX PC MS-DOS Cray X-MP Alliant	IBM VM

In an operating systems view, the simple case is the single computer system with a single operating system, represented by the lower-left corner of the matrix. The operating system in this type of architecture provides an interface between both the users and applications and the hardware. The services provided by the operating system are similar to those found on any conventional multiuser architecture (e.g., VAX). If the computer system has multiple processors, like the Alliant or Cray machines, the basic services are not changed; however additional services that allow finer control over the resources may be provided. Figure D-1 illustrates this simple version. Since both the uniprocessor and multiprocessor systems run under a single operating system, the diagram is basically the same for both.

The next layer is a single computer system that runs multiple operating systems, represented by the lower-right corner of the matrix. This scenario will not be examined in detail

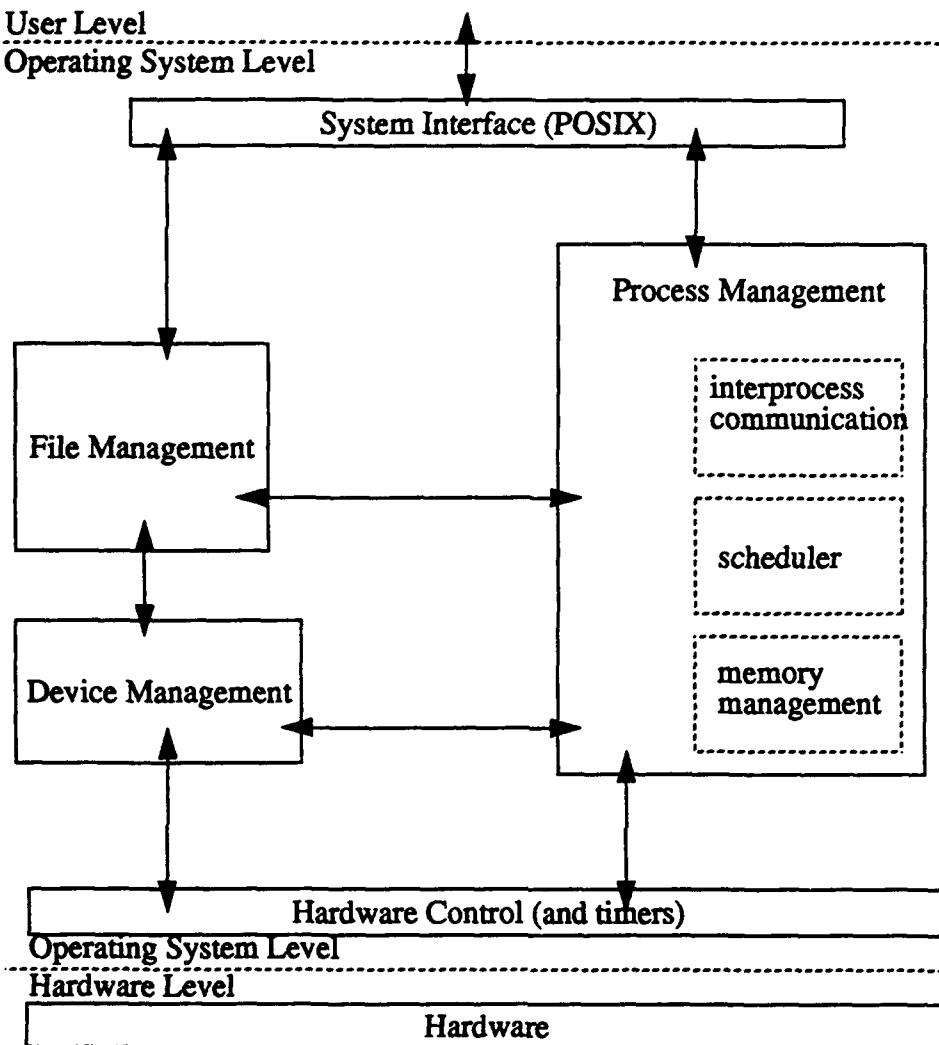


Figure D-1. Single System

because it is not a common method for system design. Examples of this type of system include IBM/VM [Buzen 1973].

The more difficult cases occur when several computer systems within a network are running together, as shown in the upper row of the matrix. In this type of network environment it is possible to have either a single operating system installed on the computer systems or several operating systems installed.

A single operating system installed on a network is often called a *distributed operating system* [Tanenbaum 1981, 2]. A distributed operating system's view of the network is seam-

less; the resources are identified by name and not by location. Thus, the system view of the network is similar to the multiprocessor's view of hardware.

Distributed operating systems are not common; however, the current trends in research are concentrating on these types of systems (e.g., Mach). The principle advantage is that the application services and system administration services are *designed* to handle a network of potentially heterogeneous systems.

The most common form of networked systems today is composed of independent computer systems, each running their own operating system. The systems are interconnected via standard data communications protocols (e.g., TCP, IP, Telnet, FTP, SMTP). This configuration is often called a *networked operating system* [Tanenbaum 1981, 2]. Networked operating systems are more difficult to use and manage than distributed operating systems because they do not provide a uniform view of the network (e.g., users must know the name and location of resources). Nevertheless, this is the current state of computer networks and will affect future systems for many years. Figure D-2 illustrates a single node in such a networked operating system.

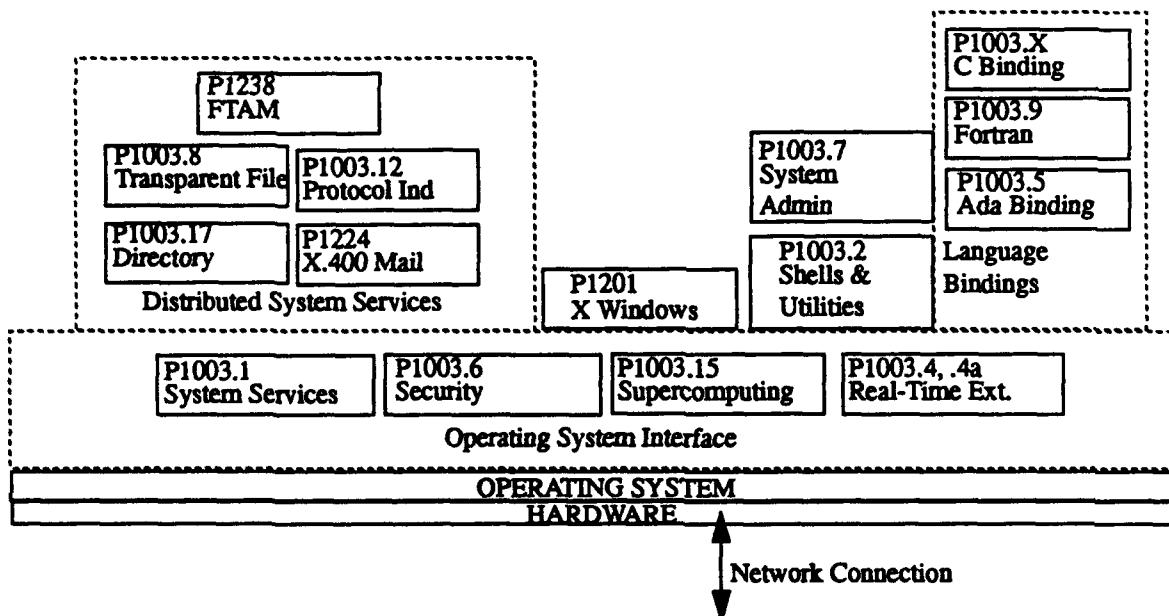


Figure D-2. A Single System in a Network (The POSIX View)

D.1.2.3 Operating System Services

There are many ways of categorizing operating system services [Bach 1986, 20-30; Madnick 1974, 8-10; Deitel 1983, 5-6]. For this discussion, operating system services are divided into five categories: process management, storage management, device management, time management, and other management functions. For each category a further division is given. Many of these subdivisions overlap. For example, communications is an important aspect of both process and device management. This is one of the reasons for the many different methods of dividing operating system services noted earlier.

Process management is a broad list of services used to establish and manage the executable processes. Process management is subdivided into four areas: process (task) management, program management, interprocess communication (IPC), and memory management. Process management controls the start-up, execution, and termination of processes. Program management assists process management in loading, executing, and overlaying the executable images of processes. IPC provides processes the functions necessary to establish communication and synchronization with other processes known to the operating system. Memory management provides virtual address mapping, memory allocation strategies, and dynamic storage allocation and deallocation.

Storage management is the part of an operating system that provides the file system and access to the file system. This involves the creation, manipulation, and deletion of files. Also provided are methods to define and access different types of files, for example, stream files, sequential files, keyed files, and the operations to read, write, update, or delete such files.

Device management involves the control of hardware interfaces to the central processing unit(s). Several types of device managers may exist in a system, for example, spooling services for printers, dedicated services for tape drives, and shared services for disk drives. A typical example of device management is the interface provided to a disk drive. Disk drives are used to store the file structure and files of an operating system and its users. The operating system provides interfaces (device drivers) to the disk drives that allow data to be read from and written to disk drives.

Time management functions of an operating system provide timers, clocks, and consistency between clocks. Timers are used to signal the passing of time. For example, in a time-sharing system the operating system uses a timer to signal the end of each timeslice. Clocks

are used to display the time to the users and mark data when it is modified or examined. Consistency between clocks is an important aspect of distributed and multiprocessing systems that is discussed later in this appendix.

Effective administration of a computer system is dependent on operating system functions not directly associated with the previous management areas. Namely, these functions are: controlling access to a system, establishing accounts for users, installing applications, creating backup copies of file systems, enforcing security, and performing configuration management.

D.2 WAM REQUIREMENTS

In this section, we list requirements specified in authorizing documents for the information needs of WWMCCS. We discuss how those requirements affect particular areas recognized as traditional operating systems areas of interest. This is done in order to limit the scope of the problem in addressing each requirement and because most operating systems research and literature can be similarly categorized.

In our discussion of operating system services, we are guided by two general requirements:

- a. The operating system features for a CCIS architecture must enable and, where possible, enhance the achievement of requirements specified for the other service categories, i.e., network, user interface, security, data management, data exchange, and programming services.
- b. The operating system features for a CCIS architecture must enable and, where possible, enhance the achievement of requirements specified for the primary applications of the CCIS, e.g., JOPES for WWMCCS.

We consider the following areas relevant to operating systems for developers of generic CCISs:

- a. Interactive, multi-tasking, multi-user processing
- b. Distributed processing
- c. Fault tolerance
- d. Real time

- e. Security
- f. System administration
- g. Ada.

Each area will be discussed in separate sections. A general capability objective will be called out at the beginning of each section. Requirements that explicitly call for a given capability will also be noted.

D.2.1 Interactive, Multi-Tasking, Multi-User Processing

A CCIS will provide an interactive, multi-tasking, multi-user environment.

Interactive operation of systems, applications, and communications is an explicit requirement, although it is assumed that this applies to some subset of the entire system. Anticipated types of interactions include interactive processing (job submissions), data query and presentation within applications, and decision-aided transactions. See Appendix G for a discussion of some of the issues involved with interactive systems.

Existing capabilities to support multi-user, multi-process environments must be maintained. Extensions to increase (functionally) these capabilities should be anticipated. For example simultaneous, multiple processes per user should be supported.

Database applications must support concurrent user transactions. The system must also meet the needs of a variety of user classes (e.g., novice and expert) and applications.

The following requirements support the need for these capabilities:

- a. Interactive processing [JOPES ROC 1983, 90 and 105; JCS PUB 6-03.10].
- b. Interactive tutorials at two levels; detailed for the novice and compressed for the experienced user [JOPES ROC 1983, 65-67, 98; JCS PUB 6-03.10, B-4].
- c. Action prompting with automated checklists, procedures, instructions, and decisions aids [JCS PUB 6-03.10, B-8].
- d. Multiuser environment as described in JOPES Increment 1 Functional Description document [JOPES ROC 1983].

D.2.2 Distributed Processing

A CCIS will provide a distributed environment.

The WAM program requires a high degree of connectivity between the CCIS components of WWMCCS. This connectivity includes not only nodes at a given (CCIS) level, but also connectivity to nodes at both higher and lower levels. Consideration must be given to connecting to the CCISs of allies, non-DoD systems (e.g., federal agencies), tactical systems, and supporting automated information systems (AISs), as well. Connectivity to relocatable and mobile nodes is required.

In general, the distribution of system resources should be transparent to users. Distributed applications and executive aids will be adequately supported, including transparent communications, when necessary.

Distributed and widely separated source databases must be allowed. Partial data degradation, when necessary, will occur through the use of decentralized data bases and multiple exchange routes. Capability for an advanced automated filing system for large volumes of dynamic, relatively discrete data storage is a goal. Data capture at the place of transaction and as close to the source as possible is necessary.

Computer conferencing and teleconferencing capabilities will be provided. A clock synchronization capability must exist.

The following requirements support the need for these capabilities:

- a. Connectivity to theater and supporting levels [JOPES ROC1983, 21].
- b. Data capture at place of transaction and as close to source as possible [JOPES ROC 1983, 110].
- c. Video capabilities to simultaneously display classified command and control related data at a number of locations within an organization or facility... [NIS ROC 1983, 5; JCS PUB 6-03.10, B-8].
- d. Database management systems (DBMSs) to accommodate distributed data bases with access to various data bases transparent to users [NIS ROC 1983, 38].

D.2.3 Fault Tolerance

The maximum degree of fault tolerance will be provided by the system, in accordance to the criticality of the application.

Dynamic and rapid reconfiguration of the system during periods of outages or degraded operations will be possible. Where necessary, automatic system regeneration will be supported. A data system monitor will be provided for control of reconfiguration and prioritization. An atomic transaction capability will be provided, and atomic information exchange between nodes will be possible. Verification of data transmission and retrieval will be possible. Where necessary (e.g., critical situations), a preformatted text message input capability with error checking shall be provided. Surplus independent communications resources will be provided and situated to enhance system reliability and survivability. Communications survivability will be in proportion to the criticality of the communications. The system will be protected from denial-of-service attacks.

The following requirements support the need for these capabilities:

- a. Network monitoring capability to rapidly cut off users during priority or degraded operations and for security reasons. The sites must be capable of dynamically and rapidly reconfiguring the system during periods of outages or priority operations [JOPES ROC 1983, 105; JCS PUB 6-03.10, B-6].
- b. Survivable communications [JOPES ROC 1983, 40].
- c. Automatic system regeneration [JOPES ROC 1983, 40].
- d. Stringent protocols to provide disciplined high speed atomic information exchange between nodes [JOPES ROC 1983, 18].

D.2.4 Real Time

There will be a real-time system or subsystem capability.

A portion of the system *may* be required to support real-time. Real-time applications are distinct from "high-performance computing" in the sense that applications which do not meet time deadlines produce *wrong* answers as opposed to simply *late* answers.

Real-time automated capabilities to receive, store, process, display, and integrate all environmental data in support of air, land and sea operations, surveillance, and other areas

(as necessary) will be provided. Event-driven processing capabilities to satisfy real-time requirements will be provided. Some data management functions will need to be reduced from 1-2 days to 5-10 minutes. Airborne command posts will interface through available communications systems to securely obtain, process, and display timely data. Voice input and output (I/O) capabilities to support selected functional areas in time sensitive situations are required. Real-time algorithms for "what if" simulation analysis are necessary.

The following requirements support the need for these capabilities:

- a. A high-density quick storage medium such as video disc capable of displaying reference documents and maps in a real-time mode [NIS ROC 1983, 10].
- b. Voice I/O [JOPES ROC 1983, 40].
- c. Real-time automated capabilities to receive, store, process, display, and integrate all environmental data in support of air, land, and sea operations, surveillance and other areas as necessary [NIS ROC 1983, 12].
- d. Event-driven processing to satisfy real-time information requirements [JOPES-ROC 1983, 65].
- e. Real-time algorithms for "what if" simulation analysis [JOPES 1983, 65 and 68; JCS PUB 6-03.10, III-8].

D.2.5 Security

The system will provide adequate security to protect sensitive and classified information in accordance to relevant directives and laws.

Full multi-level security (MLS) mode of operation for some or all of the system is a goal. Flexibility to develop incremental security solutions to meet this goal is required. Secure processing facilities for multi-level, classified information, intelligence information, and close-hold planning data will be provided. The system will be secure from unauthorized access, data manipulation, and denial-of-service attacks. Secure communication of information is required.

The following requirements support the need for these capabilities:

- a. Full multi-level security to include intelligence information [JOPES ROC 1983, 40].

- b. Incremental security solutions [JOPES ROC 1983, 89].
- c. Secure communications to DCS and NATO for both digital and analog information [JOPES ROC 1983, 90].
- d. Close-hold planning capabilities [JOPES ROC 1983].

D.2.6 System Administration

The system will be administered in a manner which facilitates all other requirements.

System accounting and performance monitoring are required. Priority operations may require the rapid and dynamic reconfiguration of the system. Communication traffic is to be managed so as to accomplish the functions relying on communication services.

The following requirements support the need for these capabilities:

- a. System accounting and performance monitoring [JOPES ROC 1983, 90 and 105].
- b. Network monitoring capability to provide the ability to monitor system activities and to rapidly cut off users during priority or degraded operations and for security reasons. The sites must be capable of dynamically and rapidly reconfiguring the system during periods of outages or priority operations [JOPES ROC 1983, 105; JCS PUB 6-03.10, B-6].
- c. Flexibility to accomplish traffic management [JCS PUB 6-03.10, A-4].

D.2.6.1 Ada

The Ada programming language will be used to the maximum desired degree for developing systems and applications software.

Ada programming language use is mandated for all DoD procurement and development. This requirement applies to all systems and applications programs which are developed by the government or government contractors. The system must provide the necessary features for the development, management, and execution of Ada programs. Specific programming services requirements are addressed in Appendix E.

The following requirements support the need for these capabilities:

- a. High-order languages; Ada is mandated [NIS ROC 1983, 38].
- b. Software transportability (i.e., Ada).
- c. Programming development environment.

D.3 DERIVED OPERATING SYSTEM REQUIREMENTS

The WAM system requirements described in Section D.2 imply the need for a wide spectrum of operating system services to support application software and overall system management. The primary purpose of this section is to expand the broadly stated system requirements into more specific operating system service requirements. Hence, the requirements described in this section are called "derived" operating system requirements. Second, the comparable services and capabilities provided by POSIX are identified and briefly evaluated.

The outline of this section follows the system requirements shown in the left-hand column of Table D-1. For each of these requirements, derived operating system services required for process management, storage management, device management, time management, and other important functions are presented. The same topics are then discussed in the context of POSIX. The effect of this section, therefore, is to fill in the body of Table D-1, which now contains the section numbers where the topics are presented.

It should be noted that fault tolerance is not addressed in this section. Some aspects of fault tolerance are covered in Appendix B on data management services (e.g., distributed transaction processing), and some are covered in Appendix C on network services (e.g., reliable communications and network management). Operating system support for fault tolerance is not approaching standardization, and is left as an area for future study.

D.3.1 Interactive, Multi-Tasking, Multi-User Processing

This section discusses WAM operating systems requirements to support interactive multi-tasking and multi-user computer systems. A *multi-tasking* system allows each user to have several processes active at once. The processor switches between the processes on an as-needed basis with the intent of maximizing overall throughput. For example, a user may run a process in the background while at the same time a process is running in the fore-

Table D-1. Requirements versus Operating System Functions

WAM Requirements	Operating System Functions				POSIX
	Process Mgmt.	Storage Mgmt.	Device Mgmt.	Timer/Clock Mgmt.	
Interactive, Multi-Tasking, Multi-User Processing			D.3.1.1		D.3.1.2
Distributed Processing			D.3.2.1		D.3.2.2
Fault Tolerance					
Real Time			D.3.3.1		D.3.3.2
Security			D.3.4.1		D.3.4.2
System Administration			D.3.5.1		D.3.5.2
Ada			D.3.6.1		D.3.6.2

ground. *Multi-user* systems have the ability to support more than one user concurrently. Thus user A and user B have the ability to run programs at the same time. A multi-tasking/multi-user system adds complexity to the operating system since it requires each process and each user to be independent from one another and to compete for limited resources.

D.3.1.1 Operating System Support for Interactive, Multi-Tasking, Multi-User Processing

Multi-user operating systems implement a collection of virtual machines, each giving a user the illusion that he/she has a dedicated machine. The virtual machines support user processes and allocate resources among the user processes. The resources managed by the virtual machine include the processor, memory, storage, and devices as presented in this section [Andleigh 1990, 3]:

- a. Process management

In order to maximize the number of users receiving acceptable response times in a multi-user environment, the operating system needs a process scheduler. The scheduling algorithm provides an orderly procedure for sharing the CPU among the operating system peripherals, user applications, and system functions called by the user applications. According to [Tanenbaum 1987, 80], a scheduling algorithm tries to achieve the following:

- (1) Fairness, make sure each process gets its fair share of the CPU.
- (2) Efficiency, keep the CPU busy.
- (3) Response Time, minimize response time for interactive users.
- (4) Turnaround Time, minimize the time batch users must wait for output.
- (5) Throughput, maximize the number of jobs processed per hour.

When resources are shared among users who maintain exclusive control over particular resources, it is possible to develop a situation where some user processes will never be able to finish. This is commonly referred to as a deadlock. Thus, the operating system must also deal with deadlock prevention, detection, and recovery mechanisms.

Memory management is a complex issue in a multi-user system because the total requirements of the users and the operating system generally far exceed the available system memory. A common solution to this problem is virtual memory. The basic idea behind virtual memory is that the operating system keeps parts of the program currently in use in main memory and the rest on the disk. Pieces of the program (e.g., pages) are then swapped between memory and disk when needed. The schemes for managing this are swapping, segmentation, and demand paging. Thus, virtual storage systems facilitate the operation of shared multi-user systems [Tanenbaum 1987, 207].

The operating system must also protect the memory from unauthorized users. In a multi-user environment the operating system must sequence access to a shared storage location so that two processors do not attempt to modify it at the same time and sacrifice data integrity [Deitel 1983, 32]. Storage protection is also essential since it limits the range of addresses a program may reference. Storage protection may be implemented by bound registers or by storage protection keys.

Thus, as a program executes, all storage addresses must be within the bounded registers or the program's key must match the storage protection keys [Deitel 1983, 29].

Applications frequently invoke a number of processes to perform different tasks or functions. The processes need a means of communication to exchange data or synchronize execution for completing the task in a required sequence. The following mechanisms should be provided by the operating system for IPC: unnamed and named pipes (a form of process used in UNIX environments), shared memory, message queues, semaphores, signals, and sleep/wakeup calls [Andleigh 1990, 94].

b. Storage Management

When several users are working together on a project, they often need to share files. It is often convenient for a shared file to appear simultaneously in different directories belonging to different users. Directories are storage structures designed to hold groups of files and programs.

The file management is more complicated in a multi-user system than in a single user system because it must associate users with files and provide mechanisms for file sharing and access control. File management is responsible for storing information on disk drives and retrieving and updating this information as directed by the user or a program. In a multi-user environment, this is commonly accomplished with one or more of the following mechanisms: virtual disks, file directory structure, file identification, file access and control routines. Refer to Section D.4.6 for more details of operating system security mechanisms.

c. Device management

In a multi-user environment, disk drives, terminals, printers, communications lines and other peripherals are shared resources (devices). The most efficient means of managing these resources is for the operating system to start a transfer to a device and attend to the device only when service is requested. This is efficient since the operating system does not spend any time making repeated checks to see if the device transfer has completed. Hence mechanisms are

needed to provide a link between the user and a peripheral device and for requesting service when needed. These mechanisms are referred to as device drivers and interrupts, respectively [Andleigh 1990, 4].

d. Time management

To prevent users from monopolizing the system, the operating system uses an interrupting clock or interval timer to take the CPU away from one user and give it to another user. This helps to guarantee reasonable response times to interactive users by preventing the system from getting hung up on infinite loops and by allowing processes to respond to time-dependent events. In addition, the operating system should provide a time-of-day clock for the computer to keep track of time in increments as fine as or finer than a millionths of a second. This time may appear relative to the user time or the system time.

e. Other functions

Another operating service not mentioned above include the ability to tailor the user's workspace to suite each individual's needs. This is achieved by the use of environment variables. For example, environment variables provide the ability to control the default settings for commands such as the print command.

D.3.1.2 POSIX Support for Interactive, Multi-Tasking, Multi-User Systems

POSIX's standard 1003.1 defines a standard operating system interface and environment based on the UNIX operating system. Since UNIX is an interactive, multi-tasking, and multi-user system, most services described in POSIX 1003.1 are relevant to this section. POSIX 1003.4 standards provide additional relevant functionality. But keep in mind that these standards describe the external characteristics and facilities that are of importance to applications developers, rather than the internal construction techniques employed to achieve these capabilities. That is, the standards define an interface, not an implementation. POSIX support can be summarized as follows:

a. Process management

- (1) To keep the processes separate, each one is uniquely identified during its lifetime by a process ID number which may not be reused by the system until the process lifetime ends.

- (2) The POSIX interface is that of a multi-tasking timesharing system; it supports scheduling with standard functions such as "kill" and "suspend" but it does not provide a scheduling policy.
- (3) POSIX has a deadlock detection mechanism; the "fcntl" function fails when deadlock occurs. But it does not support deadlock prevention.
- (4) POSIX administration functions include process creation, execution, and termination.
- (5) IEEE P1003.4a provides functions for thread management, where a thread is a lightweight process or thread of control (multiple threads can concurrently execute within a single process's address space).
- (6) POSIX does not specify memory management policies. It relies on the C library as the interface to the operating system for dynamic memory allocation. In the future, the POSIX 1003.1 group plans to develop a language-independent services specifications. The 1003.5 group is developing bindings for Ada, and the 1003.9 group is developing bindings for Fortran [Emerging 1990].
- (7) Event notification via signals is defined in IEEE 1003.1, P1003.4, and P1003.4a.
- (8) IPC via shared memory and semaphores and mutexes is defined in IEEE P1003.4 and P1003.4a.
- (9) IPC via message passing is defined in IEEE P1003.4.

b. Storage Management

- (1) User and group identification is provided by the POSIX standard. A group ID consists of one or more user and allows privileges to be awarded to users in a group. The original creator of the file is identified with the owner ID. POSIX's file control function (*fcntl*) provides for control over open files.
- (2) Files in the system are uniquely identified and organized in a hierarchical structure in which all of the nonterminal nodes are directories and all of the terminal nodes are any other type of file. Each user will have a home directory that is the initial working directory from the user database.

- (3) File management including file creation, removal and ownership. With multiple users on the system, POSIX relies on proper user identification and process identification to implement file protection.
 - (4) POSIX does not provide support for memory protection.
- c. Device Management as described in POSIX 1003.1 is limited to a general terminal interface that controls asynchronous communications ports. In essence, each user session may have at most one controlling terminal associated with it, and a controlling terminal is associated with exactly one session. The terminal is relinquished only when the session is terminated.
 - d. Timers and Clocks
 - (1) Suspend and delay process execution.
 - (2) System time.
 - (3) Process time.

D.3.2 Distributed Processing

There are two primary methods for utilizing a network of computer systems. One is a group of independent operating systems cooperating in a distributed environment. The other is a single operating system running on two or more machines. The first type is the most common type of distributed computing environment and is the type addressed by the current standards. For example, a network of personal computers running MS-DOS and UNIX. The second type is the subject of several research projects and is not addressed by the current standards. In the following discussion, emphasis is placed on the first type of operating environment.

D.3.2.1 Operating System Support for Distributed Computer Systems

In a distributed computing environment, the primary function of the network is to support the communication between systems. The operating systems, communications protocols, and data exchange protocols are used to achieve smoother cooperation between systems and applications.

Operating systems are essential to the use of distributed computer resources. In their traditional roles, operating systems manage only the local resources of a machine; however, in present networked environments, users and applications demand access to resources located throughout a network. For this reason, many applications have been developed to provide consistent communications between cooperating operating systems. Once the communication channels are open, protocols are then used to extend the resources available at individual locations in a computer network.

Thus, the operating system cooperates with other operating systems through communications protocols to provide resource management across a network of computers. The operating systems provide access to distributed storage by extending the file system structure to include file systems of remote machines. They provide configuration management facilities to control the distribution of architecture dependent files (as well as operating system dependent files) to their proper destinations. They provide message and mail routing and delivery services that allow users to communicate between remote sites. They provide access to services that are nonlocal to the users such as tape drives, application programs, compilers, and databases. They also provide file transfer, remote log in, and other capabilities.

D.3.2.2 POSIX Support for Distributed Computer Systems

The POSIX operating system interface, 1003.1, is designed for the traditional multiprogrammed environments. As a result, 1003.1 does not provide direct support for distributed computer systems; however, other POSIX standards are addressing some aspects of distributed computer systems and research in this area is still underway:

a. Process management

In distributed computer applications, the operating systems services for process and memory management are similar to those for nondistributed applications, namely, the operating system provides the facilities necessary to execute a process, provides the executing process with local memory resources, and protects the address space. Section 4.1 on multiuser and multiprocess systems gives details of these services.

In order to operate within a distributed environment, an application must coordinate its execution with other applications in the system. When the coordinating applications are located in different computer systems, a communication protocol common to both systems must be established. Appendix C "Network Services," discusses the network protocols and OSI in detail.

The coordination between applications in a distributed environment involves IPC. There are two forms of IPC, synchronous and asynchronous. Remote procedure calls and rendezvous are examples of synchronous communications, while mailboxes and sockets are examples of asynchronous communications. While P1003.4 provides both synchronous and asynchronous communications, it is not specifically designed for distributed applications. Details on networks can be found in Appendix C.

An integral part of IPC is a shared understanding between the communicating applications of the data. Since each machine in a network may have a different representation of data and data items communicated between processor may contain pointers, a data exchange protocol is needed to convert data types to a common representation and remove references via pointers. A complete discussion of data exchange can be found in Appendix A, Data Exchange.

b. Storage management

In distributed systems storage management involves several features. First, access to remote file systems is used to provide a seamless view of storage that is located on different computer systems. P1003.8 is the POSIX standard defining the interface for remote file systems. Operating systems that wish to take advantage of remote file systems must provide hooks within their own file system structure that result in calls to the network file system. Thus, the users and applications see the remote file system as if they were local to their system.

Another important feature of storage management is support for workstations without disk drives. Since all data and programs are stored on remote file systems, diskless systems require support for booting the operating systems at start-up as well as provisions for access to remote file systems.

c. Device management

The most common issue in distributed device management involves I/O devices such as printers and tape drives. The operating system in a distributed system must be capable of recognizing that a device is located on a remote system. Furthermore, the operating system must forward the user's request for service to the appropriate remote system. POSIX provides no explicit support for these concepts.

d. Time management

In a distributed system of communicating processes, time and the consistency of time are important. For example, applications that cooperate through shared services often must have the same view of time. Many applications require that time be a monotonic increasing function. These applications will not operate correctly if messages or data are received with a time stamp that is older than some time stamp that has previously been seen. For example, a file must always contain the most recent data. If an update to a file is attempted with a time stamp that is older than the latest update, then the update is in question. Clock skew is another important factor in trying to maintain a consistent view of time in a distributed system. Clock skew occurs when two machines have slightly different times. POSIX provides no explicit support for these concepts in a network; however, 1003.1 does provide time functions for individual computer systems, and other standards do exist for networks (TSS, Time Synchronization Service; NTP, Network Time Protocol).

e. Other services

(1) Authentication and authorization

Distributed systems require authentication and authorization to maintain "security." Authentication is used to verify that the request for services from a remote site is being made by a valid user. Authorization is used to verify that the valid user is allowed to make a specific request. Together these features assist in providing security.

(2) Name services

In a distributed system it is important to have a consistent view of identifiers or names. When an application on one machine communicates to another machine, the naming convention used to establish the communication must be consistent as well as the names used during communication. POSIX P1003.17 is addressing the naming issues.

D.3.3 Real Time

Generic CCISs must be capable of supporting real-time computing. In the premier issue of *The Journal of Real-Time Systems*, the introductory editorial characterizes real-time computing as computing in which “the correctness of the system depends not only on the logical results of computations but also on the time at which the results are produced” [Stankovic 1989, 6]. Thus, in real-time computing systems, timeliness is mandatory. Timing constraints are imposed by the environment in which the real-time computing system exists. Typically, the environment consists of a larger controlled system, which is in turn embedded in and affected by its physical environment. The real-time computing system is the controlling system.

For many years, real-time computing was associated with relatively small, simple, low-level, sensor-actuator based process monitors and controllers. But now, real-time responsiveness is being demanded in larger, more complex systems [Stankovic 1988, 10]:

Real-time computing systems play a vital role in our society, and they cover a spectrum from the very simple to the very complex. Examples of current real-time computing systems include the control of laboratory experiments, the control of automobile engines, command-and-control systems, nuclear power plants, process control plants, flight control systems, space shuttle and aircraft avionics, and robotics.

Generic CCISs fall into the class of large, complex real-time systems. They are representative of the next-generation real-time systems, which can be characterized as follows [Stankovic 1988, 11]:

... the systems will be more complex: They will be distributed and capable of exhibiting intelligent, adaptive, and highly dynamic behavior. They will also have long lifetimes. Moreover, catastrophic consequences will result if the logical or timing constraints of the systems are not met.

D.3.3.1 Operating System Support for Real-Time Computing

Operating systems that claim to be real-time generally offer one or more of the following categories of services and features:

- a. Mission-directed, application-directed resource management**

Over the past two decades, operating system research has been focused primarily on interactive computing. Common resource management goals have been to minimize average delay, maximize average throughput, and ensure “fairness” to competing users. While such efficiency-related and fairness-related goals may be well suited to the requirements of interactive computing, they do not adequately meet the requirements of real-time computing.

A real-time operating system must be designed in accordance with the fact that a real-time computing system exists to perform a mission. The operating system should support the mission: the resource management provided by the operating system should be neither efficiency driven nor fairness driven, but mission driven.

In particular, resource management should be driven by the time constraints of the mission, as conveyed to the operating system by the application. It is the responsibility of the application to specify resource management attributes to the operating system, and it is the responsibility of the operating system to manage all resources according to the application-specific attributes.

- b. Timely response to events**

A real-time system must maintain its integrity with respect to the state of its environment. This can be viewed as a requirement to maintain external consistency, i.e., consistency between the actual state of the environment and the real-time system’s perceived state of the environment. At the same time, internal consistency must also be maintained. That is, multiple concurrent tasks that constitute an application must have accurate perceptions of the states of one another.

If occurrences that alter the state of the environment or the system itself are viewed as events, then what is required is timely response to events. In other words, a real-time operating system should be able to respond to both external

and internal events in a timely both fast and predictable manner; moreover, it should ensure that applications can also respond to events in a timely manner, through timely event notifications to applications.

c. Predictable operating system service times

To facilitate the predictability of the performance of a real-time application, the execution times of operating system functions that the application explicitly invokes (via system calls), as well as those that it implicitly invokes, should be bounded. The bounds should not greatly exceed the means; otherwise, excessive resources may have to be dedicated to the application to assure acceptable performance.

d. High-resolution time services

A real-time operating system should provide services that make time visible and accessible to applications. For example, applications should be able to set the time, read the time, and schedule events to occur at specified times, such as at periodic time intervals.

D.3.3.2 POSIX Support for Real-Time Computing

Existing real-time operating systems fall into three classes: cyclic executives [Baker 1989], priority-driven real-time executives [Stankovic 87], and priority-driven real-time operating systems. The distinction between the second and third classes is that real-time executives are intended solely for real-time computing, whereas real-time operating systems are, in a sense, general purpose operating systems capable of meeting the demands of real-time computing. POSIX (i.e., as extended by P1003.4 and P1003.4a) falls into the class of real-time operating systems. POSIX P1003.4 arguably captures the state of the practice in real-time operating systems.

In terms of our categories of operating system services, POSIX (through P1003.4 and P1003.4a) supports real-time computing as follows:

a. Process management

POSIX supports mission-driven, application-directed resource management through the following features:

- (1) Preemptive, dynamic priority-driven processor scheduling.

- (2) Priorities, generally used to resolve contention for resources (e.g., semaphore dequeuing).
- (3) Process memory locking.
- (4) Process blocking can be controlled by application (e.g., asynchronous I/O, asynchronous message sending/receiving, conditional synchronous message receiving, conditional semaphore operations).

POSIX supports timely response to events through the following features:

- (1) Asynchronous event notification.
- (2) IPC via shared memory and semaphores.
- (3) IPC via message passing.
- (4) Threads (e.g., lightweight processes).
- (5) Priorities can be assigned to asynchronous events, messages, and threads.

b. Storage management

POSIX supports mission-driven, application-directed resource management through real-time (e.g., contiguous) files

c. Device management

POSIX supports mission-driven, application-directed resource management through the following features:

- (1) Asynchronous I/O.
- (2) Synchronized I/O (for assurance of I/O completion).
- (3) Priorities can be assigned to asynchronous I/O operations.

d. Time management

POSIX supports high-resolution time service through the following features:

- (1) Services to examine and modify system-wide timers, whose data structure representations provide for nanosecond resolution.
- (2) Services to schedule events to occur at specified (absolute or relative) times, as well as at periodic intervals.

e. Other features

POSIX supports predictable operating system service times through the following feature:

- (1) POSIX defines standard metrics and requires vendors to report the values of the metrics

D.3.4 Security

This section discusses WAM operating systems requirements necessary to support a secure development and operational system. Refer to Appendix F for additional details of system security and trusted development environments.

D.3.4.1 Operating System Support for Security

The operating system for a CCIS must provide features for a secure computing environment. While overall security requirements are discussed in Appendix F, the following will identify the specific instances of required operating system features:

a. Process management

The system must be able to securely associate a process with particular user identifiers for discretionary access control (DAC) and mandatory access control (MAC) enforcement. DAC enforcement also requires this capability for group identifiers for expected (higher) levels of security functionality. The system must use this identification information to determine whether access requests are authorized for DAC and when determining a subject's clearance for MAC. Privilege propagation should occur in an well-defined, controlled manner. (See process and file privilege flags, fork() and exec(), IEEE P1003.6.)

The system must also be able to securely associate a label representing a user's clearance to an executing process for MAC enforcement. This label must be used in a comparison with an object's (classification) label to determine whether access requests are authorized under the system security policy. MAC enforcement must take precedence over but must not invalidate DAC enforcement.

An all-encompassing, privileged subject class (i.e., superuser process) should be eliminated, without loss to administrative functionality. Security relevant processes should be protected from unauthorized invocation and interference.

Execution of a program is considered a type of access to an object (a file or object code). The specification of DAC privileges in an object's access control list (ACL) is determined by the owner (creator) of the object. Changes to an object's ACL must be possible by the object's owner. Permission specification "granularity" should allow the maximum degree of system functionality without interfering with the enforcement of the system security policy.

The system must be able to securely bind a label representing an object's classification to objects under its control. When an object is created, its classification label must faithfully retain the security level of its creator, as specified in its creator's clearance. Once set, an object's classification label may not change as a result of user actions.

IPC mechanisms require special enforcement in a secure system. MAC enforcement is necessary for the IPC mechanism, whereas, DAC enforcement may be necessary for IPC mechanism. Both DAC and MAC enforcement are necessary for access to objects in memory. Newly allocated memory resources must be free of all information from previous usage. The system must be capable of providing the necessary isolation for user, application, and system resources (processes and objects), to the extent specified in administration policies and to meet functional requirements.

b. Storage management

Both DAC and MAC enforcement are necessary for access to files. Permission to access a file is specified in an ACL, which must be securely associated with an object.

c. Device management

MAC enforcement is necessary for access to devices. DAC enforcement may be necessary for access to devices. Permission to access a device is specified in an ACLs, which must be securely associated with an object.

d. Time management

DAC privilege specification may have time bounds and constraints which are enforced by the system.

e. Other functions

Administrative functions should be defined by roles which incorporate the notion of least privilege.

The system's audit mechanism must be able to associate the relevant process (user and/or group) identifier with any record produced. Audit record(s) should contain enough granularity of information (about subjects, objects, and event) to be able to reconstruct security-relevant events adequately. Audit record files and other security-relevant data should be protected from unauthorized access. The audit mechanism should allow the "pre-selection" of events to audit based on subject ID and/or event classes. Audit records should be reviewable to system administrators. The review utility should allow "post selection" searches based on subject ID and/or event classes.

The system must provide the capability to label subjects and objects with nonaccess control information, i.e., according to policies unrelated to DAC and MAC. The label values of subjects and objects are allowed to change via (unspecified) policies, independent of MAC and DAC. Information labels describe object contents rather than access control information.

D.3.4.2 POSIX Support for Security

The IEEE P1003.6 (POSIX Security Interface) effort is currently in the process of defining a standard, secure interface for POSIX-conforming systems. Whether this standard will be both adequate and complete in the near future is an open question.

The Current POSIX Security Interface effort defines five major classes of features: DAC, audit, privilege, MAC, and labeling. The current draft does not however address distributed environments. Because CCISs are assumed to be distributed systems, additional features will need to be identified and other standards examined.

D.3.5 System Administration

This section outlines some of the basic administrative services that must be supported to operate and maintain the WAM operating system. These facilities break down into two broad categories: system management and maintenance services, and services more directly involved with supporting individual users.

D.3.5.1 Operating System Support for System Administration

a. Process management

Process management comprises two categories. Configuring the operating system and initiating and controlling system-level processes make up the first category of process management functions needed. These include initializing table settings and launching start-up processes, daemon processes, and system utilization monitoring and accounting processes. Capabilities to analyze system malfunctions and to restore these processes after a system failure are also necessary. Controlling allocation of processing resources make up the second category. These include controlling process priorities, memory working set sizes, and process execution times. The ability to kill wayward user processes is also needed.

b. Storage management

Necessary file system management and maintenance services include setting and enforcing file allocation quotas and monitoring file system utilization. File creation and access time-stamps need to be provided to support file system management, backups, and archiving. A means of clustering disk storage space (for example, contiguous storage for some files) is also needed.

c. Device management

Necessary device management services include controlling physical I/O devices such as tape drives and printers, and allocating devices such as tape drives and communications channels to processes (and revoking those allocations if necessary).

d. Time management

Timing services are needed for monitoring process execution times (i.e., for identifying wayward processes) and for system utilization accounting.

e. Other functions

General user support services are needed. These include opening and closing user accounts, controlling user quotas and access privileges, and resetting user passwords.

D.3.5.2 POSIX Support for System Administration

A separate POSIX working group, 1003.7, is addressing system administration. (Reference real-time section or P1003.4 for time management functions and reference security for access control/permission requirements.)

D.3.6 Ada

Ada application software may make direct use of services provided by an operating system. In most applications these services will be used “behind the scenes” rather than being invoked directly. In addition, there are several types of extended services such as user interface, database, and communications support that will be used by applications and which rely on underlying services provided by the operating system. The principle operating systems services employed by Ada application software and by these extended services are outlined in this section.

D.3.6.1 Operating System Support for Ada

a. Process management

Every application program will require a minimal set of operating system services to execute. These include allocating the necessary memory, loading the program code, and creating and launching an executable process (all behind the scenes). Additional memory management services are necessary to support Ada’s dynamic storage allocation.

Ada supports concurrent processing in the form of tasks. Two common techniques for implementing tasks in Ada are (1) to simulate (pseudo-) concurrent execution within a single operating system process or (2) to create a separate operating system process for each task. The first approach provides Ada compiler vendors more control over concurrent execution and requires a minimum of operating system services. The second approach requires operating system support for process creation and termination, interprocess synchronization and communication, and shared memory. However, it avoids

blocking other pseudo-concurrent tasks when an executing task is blocked by the operating system, and allows the operating system to provide true concurrent execution on multiprocessor machines.

Essentially the same tasking facilities as those needed to support the second approach to Ada tasking are needed to allow application programs to access extended system services (e.g., user interface, database, and communications services). Rather than duplicating these services for each application, shared server processes need to be supported. Ideally, application processes should be able to interact with these shared server processes (except for connection and disconnection) as if they were ordinary Ada tasks within the same application. Additional services are needed to identify, connect, and disconnect application and server processes.

b. Storage management

I/O services are also essential. Ada requires support for both sequential and direct access files [ANSI/MIL-STD-1815A, Chap. 14]. File management operations that must be supported include CREATE, OPEN, CLOSE, DELETE, RESET, etc. Sequential file operations include READ, WRITE, and an END_OF_FILE test. Direct access file operations include READ, WRITE, set position (SET_INDEX), determine position (INDEX), SIZE, and END_OF_FILE. The operating system must enable the Ada run-time system to support all of these operations.

I/O operations are sometimes unsuccessful. In Ada, unsuccessful operations must raise one of a prescribed set of I/O exceptions. The operating system must enable the Ada run-time system to detect unsuccessful I/O operations (behind the scenes) and respond with the appropriate exceptions.

Ada also supports input and output of human readable text (TEXT_IO). These services are typically provided as Ada run-time library operations using the simpler sequential CHARACTER I/O services.

c. Device management

Ada provides a very low level mechanism for controlling devices, which is intended to serve primarily operating system implementation and embedded system needs. Operating systems are generally expected to support the routing and mapping of TEXT_IO (or sequential CHARACTER I/O) operations to common I/O devices such as terminals and printers.

Such services should be required of the WAM operating system to support utility software operation. User interface, database, and communications services often assume responsibility for handling low level device operations. The WAM operating system must support these special applications and, in particular, allow them to handle device interrupts.

d. Time management

Ada requires access to timing services for delay statements and to a system clock from which year, month, day, and seconds-of-day can be derived (for function CALENDAR.CLOCK) [ANSI/MIL-STD-1815A, Sec. 9.6].

D.3.6.2 POSIX Support for Ada

a. Process management

POSIX provides all the essential behind-the-scenes services necessary for loading and executing application programs. The basic system calls are execve, for chained execution, and fork, for new process creation. Dynamic storage allocation is supported by the malloc and free system calls. POSIX supports communication and synchronization between processes in the form of "sockets." It is possible to implement Ada's task synchronization and parameter passing semantics with POSIX sockets. Some of the system calls necessary are connect, listen, send, recv, and kill.

These facilities are sufficient to support shared server processes. POSIX (IEEE 1003.1), however, does not provide a shared memory mechanism and, hence, cannot fully support implementation of Ada tasks as full-fledged operating system processes. An alternative lightweight tasking facility called threads, which provides another approach to implementing Ada tasks, however, is provided by P1003.4a.

b. Device management

POSIX provides sufficient I/O capabilities to support all of Ada's sequential and direct access file operations, including all TEXT_IO operations and routing data to common terminals and printers.

c. Time management

POSIX provides the systems calls get_timer and set_timer for controlling an interval timer for Ada's delay statements. Access to the system time-of-day clock is provided by calls to get_time_of_day.

D.4 CONCLUSIONS

The conclusions presented in this section concentrate on items of general concern and not specific problems identified in previous sections. As further analyses are performed, the conclusions and recommendations will be updated.

This appendix describes a collection of operating system services and capabilities necessary to support the requirements for a generic CCIS. Our discussion of CCIS requirements was made more concrete by considering a specific example of a CCIS, the WAM target architecture. This approach allowed us to analyze how particular CCIS requirements affected general operating system services and capabilities. This approach also allowed us to determine how the adoption of the POSIX family of standards is likely to affect the realization of these requirements.

D.4.1 Analysis Of Derived Requirements

The intent of this section is to identify potential conflicts that may arise between the operational and nonoperational requirements for operating system services. For example, operating system services needed to support security may conflict with services needed to support real-time programming. Analyses of such conflicts are impossible to conduct, however, until all the requirements are fully assembled.

Conflicting requirements can seldom be avoided in large complex systems such as generic CCISs. Two possible approaches for handling conflicting requirements are (1) to

partition the system so that requirements for individual parts do not conflict, or (2) to prioritize the requirements so that intelligent compromises and trade-offs can be made.

The first approach has been embraced by POSIX working groups. POSIX has formed a series of subgroups to define Applications Environment Profiles (AEPs) that support specific application areas. Profiles are being developed with groups of experts in each major application area who identify existing base standards and focus on applying and extending POSIX to meet their specific needs. In addition to establishing standard profiles, the AEP groups can suggest changes to the base POSIX standards and recommend creation of new standards working groups [Isaak 1990, 67-70]. The purpose of AEPs is to help specify systems that can be built or procured so that the procurement office, developers, users, and platform suppliers can communicate their needs in an unambiguous manner. AEPs are meant to simplify the software developer's task of identifying relevant standards to ensure the application is portable. System purchasers can avoid the overhead and cost of a system that provides more functionality than required. And vendors can focus on niche markets with specialized systems that implement the requisite profiles.

Since each computer system or group of computer systems in a generic CCIS may have to satisfy a different set of requirements, POSIX profiles will be beneficial in specifying these differences and in achieving a balance between conflicting requirements.

D.4.2 Overall Evaluation Of POSIX Support Of Generic CCIS Requirements

As discussed previously, POSIX is a standardization effort focused on developing standards to support open systems. The main POSIX standard for operating systems is 1003.1. In general, 1003.1 is oriented toward interactive multi-user applications. Consequently, 1003.1 lacks support for specific needs of other types of applications. POSIX, as a result, has established a set of subgroups to address these specific needs. A discussion of the standards being developed by these subgroups and their affect on the generic CCIS will be provided in the final version of this appendix.

We can state several of the anticipated benefits and potential risks likely to be encountered by users of the POSIX standards and, in areas where the risk is significant, we can make recommendations for reducing the risk and coping with deficiencies.

D.4.2.1 Anticipated Benefits

Our focus in this discussion is on operating systems and operating system services. Many of the benefits anticipated from the use of POSIX, however, are manifested in the application software. For example, POSIX provides a standard interface to operating system services and this, in turn, makes applications more portable between systems provided that the applications utilize only the standard interface. A standard interface to operating system commands and utilities also provides users with a consistent view of the system when they move from one computer to another.

Because POSIX defines standard interfaces rather than standard implementations, POSIX interfaces can be implemented on top of many different underlying operating systems¹. In addition, the POSIX standardization effort has wide support from major computer manufacturers. Thus, we predict that a variety of hardware platforms will provide POSIX interfaces to their native operating systems. This will further reduce the risk associated with the selection of a computer system, by assuring that hardware vendors provide compatible replacement systems. Wide support for standard interfaces also implies increased competition between vendors for functionally equivalent computer systems. This should control the price of equipment, increase the quality of products, and increase the number of choices to satisfy the user's price/performance needs.

Software developers are also participating in the POSIX standardization process, and their participation will increase the number of applications that are compatible with POSIX systems. Similarly, the quality, variety, and availability of software products will increase and prices should decrease.

D.4.2.2 Potential Risks

A strong commitment to POSIX also carries several risks. National and international standards bodies will not address all of the goals of generic CCISs in the WAM time frame. There are two reasons for this deficiency. First, some of the requirements presented in previous sections are not fully understood and consensus on the best, or even most reasonable, solutions has not been reached by researchers and practitioners. Thus, it would be prema-

1. Many people believe that POSIX is only applicable to UNIX-based operating systems because UNIX was the basis for 1003.1. This is not true; in fact, many operating systems provide services very similar to those required by POSIX.

ture for POSIX to standardize a particular solution at this time. Second, POSIX is a large standardization project with all-volunteer participants. Often, the participants have different preferred methods for solving a problem. As a result, the standardization process is slowed down while compromises are negotiated.

Although these two points have been phrased as risks, they also represent major benefits for standardization. In areas where solutions are not fully understood, the slow movement of POSIX reduces the likelihood that POSIX will adopt a premature, incomplete, or otherwise unacceptable solution.

Next, there is the risk that different POSIX working groups will standardize on divergent solutions. Even though the groups meet concurrently and interact, this does not guarantee consistency between working groups.

Finally, there is the risk that POSIX will not become a widely accepted standard. We believe that this risk is minimal because of the wide participation by major hardware and software vendors.

D.4.2.3 Recommendations for Minimizing Risks

The following section is incomplete in that it does not address the deficiencies that were identified in our analyses of operating system requirements. When the analyses are complete, methods to minimize risk and to work around deficiencies will be described. Currently, this section presents general recommendations for minimizing risks.

Since the WAM Program Office is making a strong commitment to POSIX, the Program Office could benefit from participation in the POSIX standards groups. There are two significant reasons for participating: (1) to keep abreast of changes to the standards and (2) to influence the standards.

The POSIX development effort is in its early stages and many of the standards are in draft form; thus, change is highly probable. Staying informed will reduce the risk of surprise and allow some time to plan and make the transition to changes in the standards. Thus, the Program Office will be better prepared to set policy and procedure for contractors and better prepared to deliver comments to the IEEE and other standards bodies during the public review process. Involvement will be especially important during the NIST public review for FIPS.

NIST is heavily involved in POSIX and it is expected that NIST will develop FIPSs and validation procedures for the POSIX standards. Since FIPSs are important to the procurement process, WAM's involvement in POSIX standardization will indirectly influence the resulting FIPSs.

Furthermore, participating in the development of POSIX standards will allow the Program Office to influence the content of these standards to ensure that they address the needs of WAM before they are approved by the standards bodies. There are three avenues of participation in the POSIX standardization process: (1) working groups, (2) balloting groups, and (3) independent reviews.

Working groups are responsible for developing draft standards. To accomplish this objective, POSIX holds open meetings on a quarterly basis. All of the 1003 working groups, as well as other working groups that fall under the POSIX umbrella (namely 1201 and 1238), meet at the same site at the same time.

Balloting groups are responsible for reviewing and approving draft standards prepared by working groups. Participation in balloting groups is open to any member of the IEEE or the IEEE Computer Society. However, it is conventional practice that members of balloting groups also participate in working group meetings or in independent reviews.

Independent reviews can be offered by any interested individual or organization by obtaining a copy of the current draft and providing comment in a form suitable for consideration by the working group.

All three methods of participation can have an effect on the quality and functionality of the POSIX standards. The goal behind participation is to influence the standards so that they will better reflect the needs of the WAM program.

Although participation in the development of POSIX standards is beneficial, it is not the only means for reducing risk. Currently there are several organizations devoted to influencing the standards process as well as providing products and product guidance to their members. Examples of such organizations include: (1) UNIX International, (2) the Open Software Foundation (OSF), and (3) the X/Open Consortium. Cooperation with these organization has the added benefit that the Program Office will have a voice in shaping products being developed by these organizations and their constituents.

APPENDIX E —PROGRAMMING SERVICES

E.1 INTRODUCTION

This appendix provides a technology framework of standards and a description of available and envisioned languages, software development environments (SDEs), tools, process models, development methods, and library support for providing programming services in WAM architecture. The state of technology for programming services to support generic CCIS architectures is described. These descriptions include discussions of the functions that will be provided by programming services, the status of open system standards and compliant products to support those functions, and restrictions that may constrain their adoption for WAM.

Programming services provide support in two distinct environments, development and operational. The development environment section concentrates on those services that will be necessary to support the development of CCIS software and provides "views" of programming services from the software developer, software integrator, and end-user groups. These views show what the general requirements are for each group. Standards and technologies in the areas of languages, tools, SDEs, methods, and reuse support, are considered as they relate to those user requirements. The Operational Environment section concentrates on the requirements for executing software in a deployed, operational CCIS environment and how those requirements may be satisfied. There is considerable difference between the current WWMCCS and the future CCIS for WAM. This is in part because the future CCIS will take advantage of modern software development methods. For that reason, this appendix presents some background material in a tutorial manner to help acquaint the reader with modern software development ideas.

The existing WWMCCS architecture consists of over forty command centers connected by a backbone network. Software for WWMCCS is written in various dialects of Honeywell Fortran, Cobol, and Assembler, and has over twenty million source lines of

code. For the most part, the development and maintenance of this code has been with Honeywell-specific tools supported by the Honeywell hardware.

Future WWMCCS architectures will comply with open system standards. Software will be written in Ada [ANSI/MIL-STD-1815A] to promote uniformity and portability among various, and perhaps yet undecided, hardware platforms. Programming services will take advantage of Ada Programming Support Environments (APSEs) and common interfaces to other SDEs and operating systems to allow an open system structure with a wide and powerful variety of tools and services.

E.2 SOFTWARE DEVELOPMENT ENVIRONMENT

This section presents an assessment of the state of technology in providing the common and expected services that users will need to support CCIS programming development efforts. We show an overall view of the general components and connectivity, list the general requirements, show the views from the users perspective, and discuss each area of services available in the development environment. In most cases, the issues associated with providing these services are listed and, if available, the future direction of these services is given. Figure E-1 shows the general components and overall connectivity in the development environment.

E.2.1 General Requirements and Views

The following general requirements are derived from Section 3 of this paper.

- a. High-order language use and interoperability The programming language Ada [ANSI/MIL-STD-1815A] has been mandated for use in U.S. military command and control systems [DoDD 3405.1, 2]. However, there is a significant investment in the current WWMCCS software which is written in Cobol and Fortran. Updated components written in Ada will have to interoperate with current components.
- b. Programming support environments. Most operating systems provide a minimum of support for the programming component of the software development effort. Ada requires its own Ada Programming Support Environment (APSE) for support and each Ada compiler vendor has its own APSE version. However,

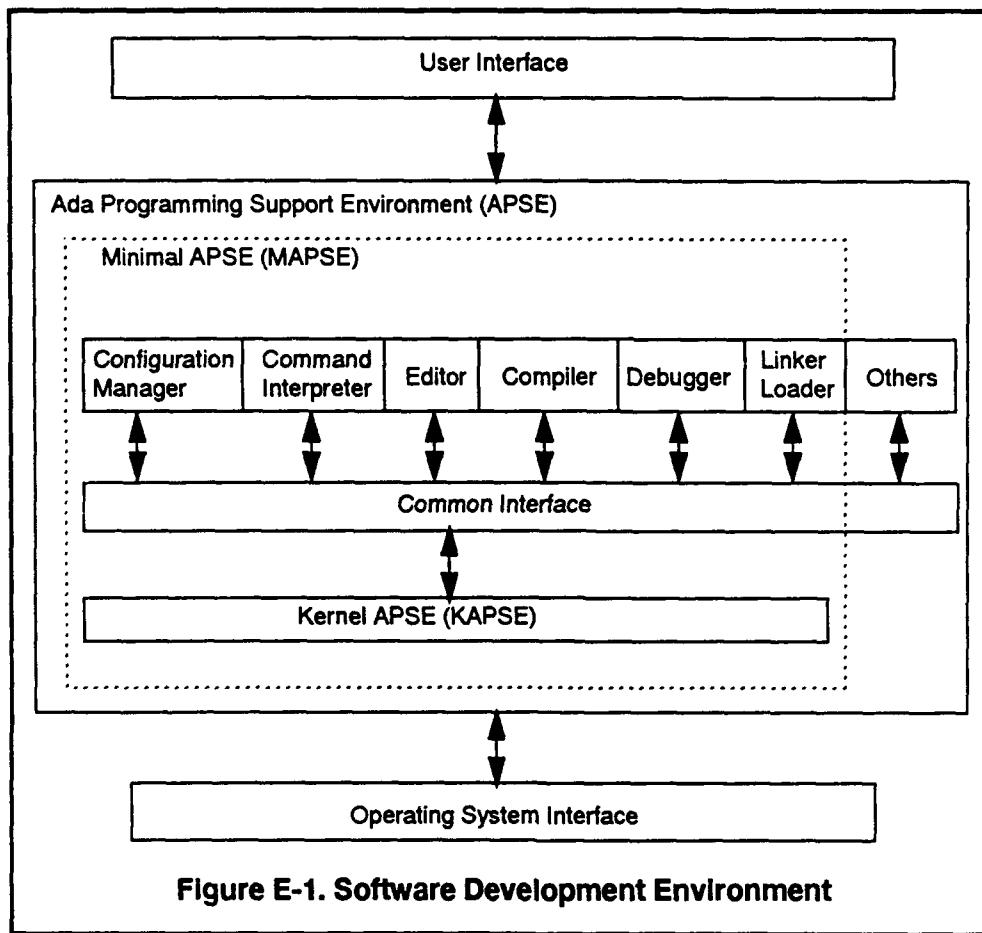


Figure E-1. Software Development Environment

there are common interfaces, such as the Common APSE Interface Set (CAIS-A) [DoD-STD-1838A] and the Portable Common Tool Environment (PCTE, PCTE+) [Thomas 1989], that will support Ada interoperability.

- c. Software development tools. The development of software is supported by the use of software tools for specifying, designing, planning, coding, testing, maintaining, and managing software. These tools must be fully integrated.
- d. Software reuse libraries. Previously developed items, including designs, modules, test cases, documentation of all kinds, as well as items specifically developed for reuse purposes, may be stored in and made accessible from software libraries to be reused in new software efforts. Expert systems capabilities.
- e. Future decision-aid software may benefit from the use of expert systems technology.

E.2.1.1 Software Developer/Integrator View

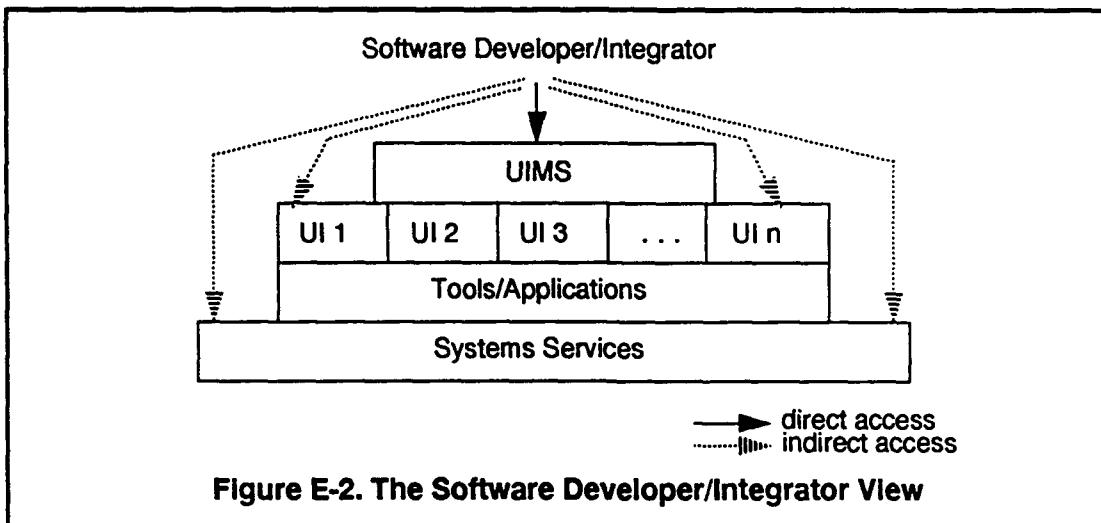
The software developer/integrator must use the facilities of the system over the entire software life cycle to develop additional tools, applications, and services for a CCIS. The software developer/integrator may also provide maintenance support for the system. The single most important requirement for the system from the developer/integrator's viewpoint is that it provide a rich, integrated, extensible environment in which software development, testing, and maintenance can occur.

General access to the system will be through a User Interface Management System (UIMS). The goal of a UIMS is to present a consistent interface to the user despite the many interfaces provided by each of the tools, applications, and services being accessed. A consistent interface will make the system easier to learn and to use and increases the portability of the user. That is, operators or users of the system may be trained only once and can be moved between sites. A consistent interface will also decrease the chance of operator error caused by confusion over which commands are to be used with which application. The interfaces of the tools, services, and applications integrated into the system will typically be general-purpose user interfaces. The UIMS will therefore need to be highly tailor able, extensible, and more robust than a simple user interface. The nature and specific requirements of a UIMS are discussed in greater detail in Appendix G, User Interface.

Software developer/integrators are concerned with invoking tools and services as part of the software development/integration process. Tool location, whether local or remote, must be invisible to the developer/integrator. When a tool or service is invoked, the location of the tools or service should not, in most cases, affect the use of that tool or service. There is an exception to this rule however. If a developer/integrator invokes a tool or service with special requirements such as special hardware support then the location of the invoking process may affect whether that invocation request can be fulfilled on a specific hardware platform. The development and integration of software for a CCIS may require teams of developers/integrators. Therefore the system must provide support for the sharing of engineering information. Similarly, the development and integration of software for a CCIS will require the use of a set of tools and services. These tools and services must be integrated so that information-sharing across tools and services is supported.

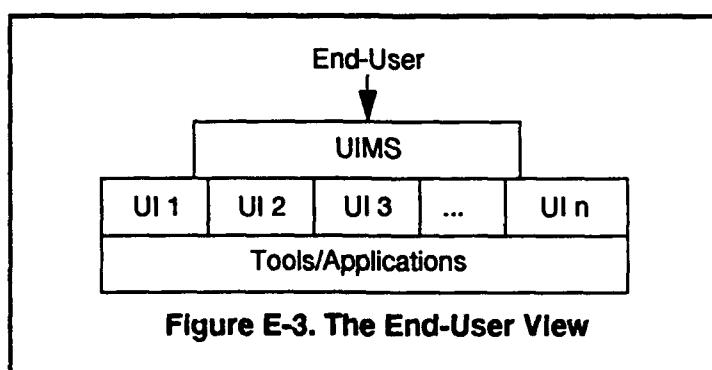
Software developer/integrators may also have access via the UIMS to the system services and hardware resources provided by the system. These services and resources may be

distributed across the system. However, to the developer/integrator, access should not be adversely affected by any given architecture. The software developer/integrator view is illustrated in Figure E-2.



E.2.1.2 End-User View

While the end-user is more apt to make use of the operational environment, access to only the top levels of the development environment is required. While the end-user is not expected to have the programming skills of a software developer, the end-user needs simple and predictable system access. By the same token, the end-user must be restricted in what can be accessed and when. The end-user view is illustrated in Figure E-3.



E.2.2 Languages

Almost all large software systems are written in a high-order language (HOL). The first HOLs, Fortran and Cobol, allowed very undisciplined, and therefore difficult to maintain, software to be produced. Tens of thousands of lines of this code still exist in WWMCCS today. Follow-on upgrades to these HOLs encouraged some structuring and modularity, but lacked an integrated approach to the entire software life cycle. The DoD's common HOL, Ada, is designed to address the entire software engineering life cycle [DoD 1978, 1]. Re-engineering all of WWMCCS ADP is an ambitious undertaking, but a transition over time using source code translation and bindings from Ada to other languages, along with the new Ada code, would lead to a system that better meets the mission needs.

Many issues concerning languages arise when discussing large software systems such as WWMCCS ADP. The general issues involve current languages and how they are processed. Specifically these issues involve:

- a. Upgrading to new language versions. What impact does this have on current software or software development efforts?

All high-level programming languages have upgrade cycles. This is one reason why all large software systems are not static. The extent to which any language changes during the upgrade cycle is dependent upon things such as:

Correcting errors. A language can be specified and designed, but when it is implemented, errors or inconsistencies in the defined semantics may be uncovered. For some languages this is not a problem; they did not become standards until after most errors were uncovered. However, for significantly complex languages such as Ada, the "discovery" period occurred after standardization. This means that software written in the current standard may not be completely upward compatible with changes forthcoming in Ada 9X.

Addressing deficiencies. The language may fail to meet its intended mission. Additional features or refinement of current features may occur during an upgrade. Software written with these languages usually has "workarounds.". An upgrade provides an opportunity to remove the workarounds.

Updating to new techniques. The state of the art is always in advance of standard languages. As new techniques come into practice, languages may be upgraded to accommodate them. A good example of this is the upgrade of Fortran 66 to Fortran 77. The newer Fortran more readily supports struc-

tured programming techniques than its predecessor. However, the older version is a subset, so upward compatibility is maintained.

- b. Language features. What effect do various language features have on system attributes?

Specific features in programming languages may affect system attributes such as performance, reliability, or security. For example, concurrency in a language requires additional overhead for process scheduling and control. This overhead reduces available resources which in turn may reduce the overall system performance. Also, some language features are implementation-dependent and are therefore implemented differently among the available compilers. Use of these features reduces the reliability as well as the portability of system software. Finally, some features may allow the programmer to unintentionally have access to sensitive data locations which may lead to a compromise of security.

- c. Compilers. Do the compilers in use conform to the associated language standard?

A significant amount of software is written using compilers that do not fully conform to the language standard. NIST conducts compiler validations for standard language compilers such as Cobol, Fortran, and Ada. Compilers for Cobol and Fortran may receive a certificate of validation even though they are not fully conforming. Validation policy for Ada states that only those compilers that are fully conforming receive a certificate of validation. Software written in Ada is therefore more likely to run on a wide variety of platforms.

- d. Do compilers provide the allowable options to maximize the language's effectiveness?

Language standards often include options for improving performance and ease of use for particular applications. Compilers can be fully conforming, but may be inadequate for the mission without certain implemented options.

- e. Translation of existing code to another language. Are there processes, policies, and tools available to assist in this?

Software may exist that would not be cost effective to re-engineer for a major upgrade. Such software usually consists of well-defined, long-unchanged algorithms written in a language that will not be supported in the future. Using a translator to convert the code automatically from one language to another is a viable option. However, policies governing the need and use of translated code, the procedures to determine which code can be translated,

and the tools to perform the translation must be in place, tested, and proven. It may be less expensive and less time consuming to re-engineer the entire system.

- f. Language bindings. What connections to other languages and tools are provided?

Some languages are monolithic, that is, they can not connect to libraries of previously compiled modules in other languages nor can they connect to tools such as windows or graphics. Languages that have bindings defined for other languages and tools allow connectivity with existing and COTS software.

- g. Documentation. Does software documentation conform to any standard such as DOD-STD-2167A?

One of the most important components of large software systems is the documentation. A complete set of documentation would contain the specifications and design through to the implementation and testing. Such documentation is vital to the upgrade and maintenance of the system. Several standards exist, such as FIPS 38 and DOD-STD-2167A.

- h. Education and training. Are software personnel adequately trained both for developing and for testing software?

Software development with inexperienced or inadequately trained personnel may lead to seriously flawed software. Even experienced programmers when inadequately trained in complex software engineering languages may produce less than ideal products. Training should cover not only programming but other aspects of software development as well, e.g., testing. A recent study by IDA of a DoD software system pointed out that personnel were partially trained to program in Ada, but received no instruction at all in how to develop, implement, and interpret tests for the code they were to write [Akin 1990].

- i. Testing. Are software testing conventions established and followed?

Testing aspects of the software life cycle are critically important. Testing is not a phase that takes place just before deployment of the system. It must be carried out in parallel with other software development activities. As soon as a software development has been completed, the work products must be tested informally by the software professional responsible for that activity. Then the work products must be subjected to formal testing by an independent software quality assurance group. Conventions and processes for all testing activities must be established and followed.

E.2.2.1 Current Standards

WWMCCS ADP software is currently coded in several languages and dialects of those languages as listed below:

a. Cobol

Cobol was developed in the late 1950s to provide a programming capability for applications that were business oriented, such as inventory tracking, forms preparation, and report generation. The language was first standardized in 1968 and has been updated twice. The three standards are: Cobol 68 [FIPS PUB 21], Cobol 74 [FIPS PUB 21-1], and Cobol 85 [FIPS PUB 21-2]. During the first phase of WWMCCS ADP construction Cobol 68 was used, but the majority of the system was written in Cobol 74. Cobol 68 is a subset of Cobol 74, so the older code has been re-compiled using the Cobol 74 compilers and can be considered to be Cobol 74 code. The latest version of Cobol, Cobol 85, has been used for extensions to WWMCCS ADP, but the majority of the system is still under Cobol 74.

b. Fortran

Fortran was developed in the late 1950s to provide a high-level capability for programming scientific and mathematical applications. Fortran was standardized in 1966 by the American National Standards Institute (ANSI). The language was revised in 1977 and was standardized in 1978. The two standards are: Fortran 66, ANSI X3.9-1966, and Fortran 77, FIPS PUB 69. WWMCCS ADP contains mostly Fortran 66, but applications and extensions developed after 1978 are in Fortran 77.

c. Ada - ANSI/MIL-STD-1815A, ISO 8652

Ada was developed in the late 1970s to give the DoD a single, common high-order programming language with which to build reliable, portable, and more cost-effective software. Ada is an ANSI, ISO, and military standard. The 1983 version is the current standard. Ada has not been used extensively in WWMCCS ADP, but those portions that were developed under the WWMCCS Information System (WIS) program were written in Ada.

d. GMAP. Honeywell's General Macro Assembler Program.

High-order languages (HOLs) like Fortran and Cobol were developed largely because Assembler code is so difficult to maintain. In addition, Assembler code is essentially non-portable and non-reusable.

E.2.2.2 Languages for a Future CCIS

Future modernization efforts for WWMCCS ADP will rely on re-engineering of the software of the current system. These efforts will emphasize modern software engineering principles and techniques and will make use of high-level tools and languages for specifying, designing, and developing software.

E.2.2.3 Ada 9X

The Ada 9X program is currently underway to revise ANSI/MIL-STD-1815A to clear up inconsistencies with the semantics and to refine certain features better to address the applications to which Ada is applied. Developing new systems or re-engineering old systems into Ada will enhance portability and maintainability while promoting uniformity. The specifics of Ada 9X are unknown at this time, as the revision process is in its early stages. Ada 9X is expected to be released in 1995.

E.2.2.4 Knowledge Representation Languages

There are currently no standards for knowledge representation languages or notations. In the absence of such standards, the architecture of the CCIS should be constructed so as not to preclude the use of expert systems in the future. For example, the development of expert system software for use as decision-making aids is promising. A CCIS architecture that allows for such aid is desirable. Prolog and Common LISP are two languages associated with knowledge-based software. ANSI committee X3J13 is working on drafts for Common LISP and Prolog standards.

E.2.2.5 Other Languages

COTS software used in the future CCIS may be written in almost any language. Bindings between Ada and other languages will in general support this software; however, there may be problems if the software requires special support or there is some need to re-compile its source. The use of SQL, fourth generation languages (4GLs), and other languages to describe and design software will also have places in the future CCIS. The extent of use

of these languages, other than SQL, in the future CCIS is indeterminate at this time. More discussion on SQL may be found in Appendix B, Data Management.

E.2.3 Environments

A well-designed software development environment can lead to increased productivity and software quality by providing a structure within which advanced software development tools and services can reside. These tools and services may simplify the development process by automating certain steps, by providing built-in checks to ensure certain processes, policies, or standards are being adhered to, or by providing decision support for the engineer. A software development environment can also assist engineering teams working on large projects by providing mechanisms to share and protect ongoing work. Three issues of concern in the design of a software development environment are the degree to which the environment provides an extensible framework allowing evolution over time, the tailorability of the environment based on the application domain, and the degree to which the environment integrates the tools and methods needed by the users.

The architecture of the environment will affect its maintainability, reliability, extensibility, and performance. An architecture that presents a centralized, monolithic set of tools and services may be less extensible than one that presents a distributed, loosely coupled set of tools and services. At the same time, unless the environment is fully integrated, it is difficult to enforce the processes and standards to be used in developing the software.

E.2.3.1 Current Standards

There are currently two significant standards efforts underway in the area of software development environments. These are the CAIS-A and PCTE projects.

The Common Ada Programming Support Environment (APSE) Interface Set (CAIS) project was begun in 1982. The goal of the project was to ensure better development and maintenance environments for DoD mission-critical computer software. A DoD mandate required that all DoD mission-critical software shall be developed using the Ada programming language. However, software development environments supporting the Ada programming language are still evolving. This provided an opportunity for the DoD to have a positive influence on the nature of the environments being developed. The Ada program-

ming language does not require special capabilities in a development and maintenance environment, but the life-cycle maintenance of mission-critical software does. The CAIS team believed that an integrated software development environment with superior tools would increase the acceptance of the language.

The CAIS is a set of interfaces through which APSE tools can access the operating system services provided in the host environment and communicate among themselves. Typically a tool that uses services provided by a given host is host dependent. Data or files generated by that tool will follow the conventions of the host. By standardizing a set of interfaces supporting a wide variety of hosts, tool portability is enhanced. This standard set of interfaces does not provide every operating system facility, just those most common and useful or necessary to ensure tool and tool database portability. In addition, the CAIS provides an object management system to ensure data integrity. The CAIS interfaces are specified as a set of Ada package specifications for services such as process control, file management, and device control.

The APSE is a collection of tools used over the entire software life cycle. As Ada was mandated by the DoD for use on all mission-critical software, the APSE must be high quality, available at many sites, and support a wide variety of hosts. At the center of the APSE is the Kernel APSE (KAPSE). The KAPSE provides runtime access to operating system services commonly available to all tools and applications. Although typically implemented differently, these services and the capabilities they provide are considered standard across hosts. On top of the KAPSE is the Minimal APSE (MAPSE). The MAPSE is a minimal toolset to support software development. These tools are written in Ada and are portable as they use the common interfaces to the KAPSE. On top of the MAPSE is the APSE. The APSE consists of project-specific tools and services.

The Portable Common Tool Environment (PCTE) project was begun in 1983 by the European Strategic Programme for Research in Information Technology (ESPRIT) [Thomas 1989]. This project was strongly influenced by the Stoneman report [DoD 1980]. The goal of the PCTE project was to describe and prototype tool interfaces that could be used to define a software development environment. The environment would comprise a set of public tool interfaces and a data management system. As defined by the PCTE project, a public tool interface is a non-proprietary interface existing as a library unit which may be used by a tool to provide access to operating system services. Tool builders might use the interfaces either to integrate or attach their tool products to an environment. The distinction

between integration and attachment reflects the degree to which the environment monitors, controls, and makes use of the information on a given tool. An integrated tool makes full use of the services provided by the environment such as logging an audit trail and data management. An attached tool does not. For instance, data is maintained in a repository known only to that tool.

The criteria for the development of PCTE were that it be policy and mechanism independent, support a distributed environment, provide easy tool integration, provide a complete interface definition, and provide multilingual support. To accomplish this, PCTE defines the services needed by the tools. The services provided by PCTE include data management, tool execution and communication, distribution and environment management, and programmer interface for user interface management. Several environments are currently being developed based on PCTE. A highly secure version of PCTE, PCTE+, is also being developed. The Portable Common Interface Set (PCIS) is an effort aimed at converging CAIS-A and PCTE+.

In addition to CAIS-A and PCTE/PCTE+ another effort for a standard tool environment is becoming organized. A Tool Integration Standard (ATIS) is an effort led by Digital Equipment Corporation (DEC) and is based upon a commercial product developed by Atherton Technologies, Inc. ATIS has been formally proposed by DEC to the ANSI Standards Committee X3H4. Support for ATIS from other corporations has been inconsistent, but DEC and Atherton remain major players. The future of ATIS is uncertain, but interaction between the ATIS group and the PCTE group may suggest that there will be a mutual evolution of the two standards.

The IEEE task force on professional computing tools has drafted a proposed standard, IEEE 1175, for tool interconnections.

E.2.4 Tools

Computer-aided Software Engineering (CASE) is a means of integrating methods and tools to provide an environment that facilitates the development of software and systems. Tools within such an environment are often referred to as CASE tools. COTS software is the primary source of software engineering tools. Very few environments consist of tools that were built entirely for a specific development effort. Proprietary tools are typically built for a specific in-house application and are never marketed outside the user group.

A software development environment without a reasonable set of tools is of little value to a software engineer. The issues of concern when populating an environment with tools are consistent user interfaces across tools, tool integration within the environment, tool interoperability, and tool coverage for the particular problem domain.

Tools are integrated into a development environment to facilitate the software development and maintenance process. For a software developer to be most productive, he or she must be able to use all the tools available. Each tool is generally designed and built with its own unique user interface. This creates a problem for the software developer. He must learn numerous user interfaces and always be aware of which tool is executing. If all the tools in an integrated environment present a consistent interface to the users, productivity will be enhanced. Tool integration reflects the degree of coordination and cooperation between a tool and the development environment. Until recently tools were often built as stand-alone entities. Since it is unlikely that a single tool vendor will produce every tool needed by a development group, tools were custom built and procured from different sources. However, the complex task of software development requires that the tools be brought together to work in a coordinated, integrated fashion. Often this integration task is left to the users of the tools. A framework or environment is necessary to enable the integration of tools.

In addition to facilitating tool integration, a framework or environment will provide a standard interface to which tool vendors can build. A standard set of tool interfaces implies greater portability for any tool built to the standard interface, and thus a greater potential market for tool products. As the market increases, both the number and quality of the tools being produced will increase.

Portability applies to the tool, the output produced by the tool, and the user of the tool. Tools built to a standard tool interface will be portable among development environments. Tools that enjoy widespread use will be better documented, tested, and supported. Tool user portability will be enhanced if tools in an environment are designed to provide a standard interface. One of the reasons that a framework for tools is so desirable is to facilitate data interchange among the tools in the environment. A framework can provide the interface and data format conventions required for tool interoperability. Rather than require that the output of one tool be manually translated and directed as the input of another tool, the goal is to allow the tools to share automatically data and other information as needed.

Tools frequently generate output that is stored in a proprietary database known only to that tool. The structure and contents of these tool databases are critical to the ability of tools

to share data. Typically, each tool will generate output in its own proprietary format. Thus, the output of the tools is locked into a repository and is very difficult for other tools to access. A great deal of valuable information may be captured or generated by the tool, but it may now be inaccessible. A better model would be for tools to read and write common databases.

As a way of describing tool coverage, Wasserman [Smith 1990, 15] has proposed a tool classification scheme based on the phases of the life cycle during which the tool is used and the functional nature of the tool. Tools may be classified as being either vertical or horizontal. Vertical tools support a single phase of the life cycle, while horizontal tools are used across life cycle phases. Examples of vertical tools include front-end CASE tools (tools for requirements analysis, specification, and design), requirements tracing tools, code generation tools, testing tools, and re-engineering tools. Horizontal tools may be further partitioned based on their functional characteristics. Tools that support individual or team project efforts include project planning tools, cost estimation tools, tracking tools, and documentation and publishing tools. Tools that support the infrastructure of the development effort include configuration management tools, version control tools, security tools, and project database management systems (DBMSs).

In addition to procuring or building tools for the development environment, an adoption strategy including training methods and standards for tool use must be developed before the most effective use of tools can be made. Tool maintenance must be planned as the tool set matures and evolves over time to meet the changing needs of the users.

E.2.4.1 Current Standards

There are currently no tool or toolset standards.

E.2.4.2 Tools for a Future CCIS

Tools that may be included in a CCIS will support:

- a. Requirements capture and tracing
- b. Specification
- c. design

- d. Development
- e. Reverse/re-engineering
- f. Performance analysis
- g. Testing
- h. Software validation/verification
- i. Software analysis (metrics)
- j. Graphics tools/user-interface development
- k. Configuration management
- l. Planning, scheduling, and monitoring
- m. Cost estimation
- n. Documentation generation/management

E.2.4.3 Expert System Tools

The future CCIS may be able to benefit from the application of expert systems in areas such as general software development, engineering decision support, development environment management, repository cataloging and retrieval, system performance optimization, resource and network configuration, executive planning and scheduling, and data analysis. Expert systems may also enhance database technology to provide expert databases. Expert databases are a relatively new technology combining the reasoning power of expert systems and the data access capabilities of database management systems. This technology is expected to have a significant impact on large-scale, distributed information systems. Other possible applications for expert systems include software and hardware diagnostics within the CCIS, and tutoring systems in the use of the CCIS. Tools may also be needed to maintain and verify the contents of knowledge bases.

Standards do not currently exist for expert system shells or knowledge representations. Nor are they likely to be available for some time. The first area of standardization for expert systems will be the interfaces between the expert systems and the rest of the system. Thus, standard bindings between expert systems and programming languages, expert systems and databases, and expert systems and user interfaces will evolve first.

E.2.5 Process Model and Development Method

Boehm [1988] defined the software process model as the ordered sequence of activities that occurs during the course of software development. The process model describes not only the specific activities that occur at each stage, but also the criteria to be used by the developer to conclude one stage and begin the next. Examples of software development process models include code and fix, waterfall, rapid prototyping, evolutionary development, phased refinement, spiral, and transform. Boehm defines a software development method as the way the specific development activities are actually carried out by the developer. A software development method will describe notations and representations used, the way functions are partitioned and resources allocated, and data and control information. An understanding of both of these views of software development is critical to the success of the product. An appropriate process model and development method can increase programmer productivity as well as increase the quality of the software produced.

The code and fix model of the software development process defines an ad hoc, but very typical approach to the problem. Using this model, code is written and then fixed as faults are found or the requirements become better understood. This sequence of activities continues almost indefinitely. The code and fix process model places minimal emphasis on the analysis of requirements. This process model also places minimal emphasis on the design of the code. As a result, the code exhibits increasing disorder as time goes on. Finally, because the code is not designed with its eventual testing and modification in mind, the fixes that are necessary become expensive to implement.

The waterfall model of the software development process defines an orderly sequence of steps or stages and feedback loops between successive stages to correct any faults in the previous stage that may be detected during the current stage. The model has been extended to support incremental development, parallel development, and risk analysis and validation at each step. The major strength of this model is that it is document driven. The criterion for completing one step and moving to the next is document completion. This approach may not be appropriate in many cases such as highly interactive applications where user feedback is not facilitated through the use of documentation. As a consequence, the resulting product may well be what the sponsor requested, but may not meet the sponsor's real needs. The weakness of the waterfall model is the written specifications document; it is extremely difficult to visualize the final product solely on the basis of such specifications.

The rapid prototyping model of the software development process overcomes the weakness of the waterfall model. The requirements analysis phase is replaced by the construction of a prototype. The sponsor and users can interact with the prototype to determine if it satisfies their real needs. In addition, information obtained from developing the prototype can assist with the specifications, design, and implementation phases, thus reducing the need for feedback loops between successive stages.

The evolutionary development model of the software development process defines a series of stages during which the operational capabilities of the product are expanded as the understanding of the problem increases. This model is similar to the code and fix model in that there is no advanced planning for what the completed product will look like. For this reason, it is not possible to guarantee that the product will be easily evolved, if at all. This model of the development process may not result in an optimal product as many important decisions such as the architecture and usage characteristics are not necessarily made a priori, but may occur very late in the development process.

A phased refinement model of the software development process defines a series of implementation phases during which prescribed design details are added to the product. However, unlike the evolutionary development model, all system functionality is specified during the first step of the development.

The spiral model of the software development process defines a sequence of activities that is repeated for each phase of the development process. Movement through the spiral represents progress through the sequence of activities of each phase. The spiral model is risk driven in that the activities undertaken at any point during development depend on the perceived risk. The spiral is begun by identifying the goals of the development process. Each of the strategy alternatives that may be selected to meet the goals are defined. For each alternative any constraints, such as cost, safety, reliability, or security are described. Each alternative and its constraints are then evaluated relative to the goals. The risks associated with the implementation of each alternative are assessed, and a risk management plan is developed. Examples of approaches to resolve the risks include simulation, prototyping, analysis, and benchmarking. At this point in the development process, the spiral model can accommodate all the other models as special cases. That is, the spiral model supports any method of development that is appropriate.

The transform model of a software development process assumes some capability to convert some formal specifications automatically into a program. The program may be

enhanced by optimizing the transformation step by providing specific guidance. At some later stage, the formal specification may be modified, and the transformation and optional optimization step repeated. This model implies that the design and code steps in a traditional model are skipped entirely. Furthermore, in this model the code is never modified or fixed directly. Rather, the formal specification is changed. A significant current limitation on the use of this model for software development is the lack of availability of automatic transformation tools for the application and hardware of interest.

E.2.5.1 Current Standards

There is currently a single standard, DOD-STD-2167A, *Defense Software Development Standard*, for the process of software development. This standard allows tailoring to any model and supersedes DOD-STD-2167 which was tied to the waterfall model.

E.2.6 Reuse Support

Reuse is a strategy with the potential to increase software productivity, reliability, and quality. However, despite a significant amount of ongoing research over the past several years, reuse is not practiced as widely as would be expected. Inhibitors include the technological barriers, financial barriers, and psychological barriers. Technological barriers include the fact that representation technologies, reuse tools, environments, and methods do not currently support reuse. Financial barriers include the significant capital, intellectual, and time investments required to construct the reuse repository before the reuse strategy pays off. A project budget does not typically include reuse library costs or the cost of making all project software reusable in future products. Psychological barriers include the “not invented here” syndrome. Engineers are often distrustful of designs or code they did not develop.

Reuse incentives include strong support from management, contracting or otherwise building the costs of reuse into the developed software, education, and strong tool, environment, and method support.

Methods or techniques for reuse include the building of object-oriented reuse libraries and retrieval schemes to assist users in extracting the appropriate item from very large libraries.

A reuse strategy can be applied at several phases of the software life cycle including specification, design, implementation, integration, testing, and maintenance. Design reuse provides perhaps the greatest opportunity for software engineers. Currently, design representation remains a major stumbling block for efforts to reuse software designs. While a programming language may be too specific for the expression of design information, a more general notation may be non-machine processable and require a great deal of manual activity.

E.2.6.1 Reuse Library Issues

The operational issues surrounding software reuse include finding a component, understanding that component, possibly modifying the component, and composing a larger structure using the component. Each of these will be discussed in terms of a range of possible approaches. The components residing in a reuse repository must be classified to facilitate the software developer finding and evaluating the applicability of the component to the task. Some type of taxonomy, possibly a hierarchy supporting inheritance, may be appropriate for the repository structure. Tools and an associated environment must be provided to allow browsing and query capability against the repository. Once found, the component is evaluated by the software developer to determine whether the entity is suitable. This task requires that the developer develop a computational model of the component. Until the developer understands the component, he or she cannot fully evaluate its potential for reuse. In an effort to evaluate the component, a HyperText system may be extremely useful to organize and access related information on the component.

The repository should support both functional and qualitative evaluations of components. The functional characteristics of a component include its attributes and some representation of what it does and how. The qualitative characteristics of a component include some objective and subjective metrics indicating for instance, the reliability of the component, its maintainability, or its performance characteristics. Other characteristics to be considered of any component located in the repository include whether it is an exact or a close match to the requirements, and what the operating system, hardware, runtime library, data, and interface requirements are. Issues such as the maintenance record associated with the component, the adherence to standards or policy, and the developer reputation may also be considered.

Rarely will reusable components be used without any modifications to form the new program. Therefore, the development environment must support tools to assist the developer in modifying the reusable components. This toolset may overlap with the general software development tools. After locating, evaluating, and possibly modifying components from the reuse library, the developer will compose a new program. Two techniques for developing new programs from reusable components are generally recognized: composition and generation. In composition, the components remain, for the most part, unchanged in their reuse. Examples of this technique include the use of subroutines and functions, or the use of components as building blocks to form new programs.

Education is critical for reuse to occur. Developers must be trained in the methods to select the appropriate components for reuse, trained to design software that is reusable, and trained to appreciate the value to be gained from a reuse strategy. To be effective, reusability must be engineered into a product from the beginning and component reuse must be considered at specification time and again at design time.

E.2.6.2 Current Standards

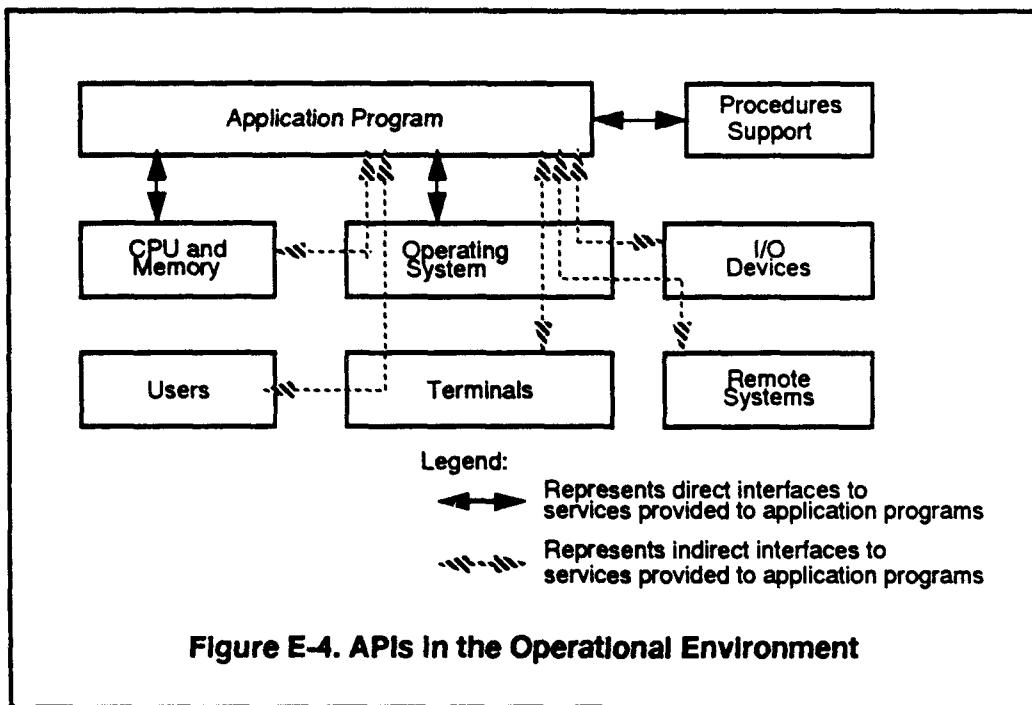
No standards have been developed in this area.

E.3 OPERATIONAL ENVIRONMENT

This section describes the elements of the operational environment. That is, it will describe the CCIS as seen from a running (or ready to run) program. Since Ada has been mandated for use in military command and control systems, the discussion of an operational environment will focus on the use of Ada applications.

E.3.1 General Description

The software developer sets out to create an application that executes on the underlying hardware. During execution the application makes use of the abilities of that hardware. Figure E-4 illustrates the relationships among the components of the operational environment. The application program has direct interfaces to support procedures, the operating system, and the central processing unit (CPU) and memory. Support procedures represent, for



example, common mathematical routines, I/O, and other common modules used by multiple applications. The operating system provides services such as process management, interprocess communication, memory management, and device and file management. The application makes use of these services through system calls. The CPU and memory represent the bare computing resources available to the application at the machine level. The software application has indirect interfaces to I/O devices, remote systems, terminals, users, and the CPU and memory. These indirect interfaces are provided through the operating system. I/O and device management services provided by the operating system enable an application to interface with I/O devices. Interprocess communication services provided by the operating system enable an application to interface with remote systems. Memory management and program and process management services provided by the operating system enable the application to interface with CPU and memory resources.

E.3.2 Ada View

In an effort to allow more effective use of the hardware capabilities, operating systems have been developed and are typically layered on top of the underlying hardware resources. Applications written in Ada are somewhat different from those written in other program-

ming languages. The Ada programming language was designed with the intent that it would be used in the development of embedded systems as well as for non-embedded applications. The Ada programming language and development environment include features specifically to support embedded systems which have unique constraints such as code size. To minimize the amount of code resident on the hardware, embedded systems typically have a minimal executive rather than a complete operating system. In some cases an Ada program must execute on a bare machine, that is, one without an operating system. Therefore to support embedded system development in Ada, all services potentially required by the application must be provided by the Ada runtime system (RTS) rather than relying on an operating system. The RTS corresponds to the support procedures in Figure E-4.

The services required by an Ada application may be divided into two classes: those affecting the generation of the code and those affecting the partitioning of runtime functions between code and runtime routines.

The services that affect the generation of code include dynamic memory management, exception management, and tasking management. In addition conventions such as the addressing models for representing pointers and data structures, definitions of predefined types and mechanisms for subroutine calls, parameter passing, and register usage across calls also affect the code generated. The allocation of the runtime functions to code and runtime routines is primarily influenced by the desired performance and the capabilities of the target configuration. The major drivers are the tasking constructs, memory management functions, exception management functions, Ada attributes, and commonly called routines such as string operations and multiword arithmetic.

Thus, the capabilities provided by the operating system, support procedures, or both form the operational, or when speaking of Ada, the runtime environment for the application. The runtime environment provides the development engineer with a more abstract view of the bare underlying computer resource. It provides the notion of an abstract machine. The concept of an abstract machine allows more effective use to be made of the underlying resources in that the engineer need not create software for all the functions required by the application. This allows applications to be created in less time and for less money. However, the support to the application may be more general and thus suffer performance degradation. The existence of a runtime environment does not preclude an engineer from directly accessing features of the bare machine from within an application. However, such direct access will adversely impact portability, maintainability, and reus-

ability. The basic elements of a typical runtime environment comprises data and code conventions as well as pre-written subroutines. Code and data conventions describe the data structures to support programming languages, mechanisms for I/O, macro generators to reproduce common code sequences, how the set of registers will be used, and how subroutines will interface with one another. Pre-written subroutines generally provide I/O and mathematical functions.

It is through the selective use of predefined subroutines and the use of data structures that the developer creates an abstract machine unique to the application. The use of the predefined subroutines acts to extend the capabilities of the machine. Programming language compilers in conjunction with the operating system provide a mechanism to automate partially the generation of the appropriate runtime environment. In general the operating system provides a predefined set of subroutines to use the underlying hardware resources, and the compiler provides the data and code conventions as well as the interface to the common subroutines. The Ada RTS may provide any additional or alternate (optimized) predefined subroutines.

The services required by an executing program depend to a large extent on the high-level language in which the program is written. The capabilities provided through the language require the support of the underlying machine. The exact nature of the support varies among languages. Through the compilation process, code is generated and files are linked to form an executable image of the program. This program then relies directly on the operating system, runtime system, or underlying machine for further services.

E.3.2.1 Operating System Services

The services an application may receive from the operating system include memory management, interrupt management, I/O management, and multitasking (process management) support. For example, the services available on a UNIX-like operating system through POSIX include those from Appendix D, Operating System Services, shown in Table E-1.

Table E-1. Operating System Services and Functions	
Service	Functions
Process Management	Create and Terminate Get Information
Program Management	Load and Execute
IPC	Signals Pipes
Memory Management	-none-
Storage Management	Create and Delete Open Synchronous Access Get and Set Information Directories
Device Management	Generic (as files) Terminals
Time Management	Delay Alarms
Event Management	Signals
Other Services	Password Encryption System Data Access

E.3.2.2 Ada Runtime Services

The support required by an application implemented in Ada may be roughly categorized as either memory management, control management, or binding management [Freedman 1985, 157]. The specific functions provided to an Ada application in each of these categories is described below.

a. Memory Management

The Ada programming language allows the creation of objects at runtime. As a result, following compilation and the static allocation of storage, some mechanism is needed to allocate storage dynamically during program execution. A function in the RTS, the *Dynamic Memory* function, is responsible for supporting the capability of the Ada programming language to create objects during execution. Two routines associated with this function, the *Stack Management* function and the *Heap Management* function, are used. These functions are supported by a predefined exception handler routine should the *Storage Error* exception be raised. Implementations of memory management functions such as stor-

age reclamation are non-standard in the Ada language. That is, garbage collection may or may not be provided by the RTS. This implementation-dependent service impacts the speed of execution, timing of code, and data storage capacity.

b. Control Management

Control management makes up a large part of the services provided to an Ada application. In addition this area of a runtime support system is the most critical in terms of performance. The functions of control management include exception handling, process management, and I/O support.

Exception Handling functions handle both predefined and user-defined Ada exceptions. The predefined exceptions are those provided by the RTS. The exception management function is invoked whenever an exception is raised, either by the RTS or the user program. The function identifies a corresponding exception handler, and transfers control to it. If no handler exists, the task or main program is terminated. During program execution the RTS performs a series of checks on the status of the program and its environment. If an erroneous condition is detected by the RTS it can raise the appropriate exception. The RTS provides a library of exception handlers to process these exceptions during execution. The predefined exceptions supplied and the checks performed by the RTS include those listed in Table E-2. Performing elaborate checks at runtime consumes resources. This may adversely effect the execution of the Ada program. Ada provides the mechanism to suppress the checks. However, an implementation of the RTS may ignore a request to suppress the runtime checks. Interrupt Management support provided by the RTS allows an executing Ada program to respond to asynchronous events such as machine interrupts, UNIX signals, and system traps.

The *Process Management* function allocates processors among the processes ready to run. In Ada, tasks as well as main programs may be viewed as processes. It is invoked by other functions in the runtime environment to coordinate the allocation of resources available in the system. If processor resources are not available, the process management function will cause any ready tasks to become blocked. As resources become available these tasks are unblocked and are assigned to processors based on their priorities.

Ada tasks are created dynamically at runtime. The runtime environment must support the efforts of the application to create, destroy, and communicate among its tasks. A task is

Table E-2. Checks Performed on Ada Predefined Exceptions

Exception	Check Performed
Constraint Error	Access Check Discriminant Check Index Check Length Check Range Check
Numeric Error	Division Check Overflow Check
Program Error	Elaboration Check
Storage Error	
Tasking Error	

created by allocating space for the objects representing the task. The task activation function is then invoked to execute the declarative portion of the task, thus activating the task.

A task execution may communicate with another task using the Ada rendezvous construct. All synchronization and communication between tasks is provided for by the Ada language. The constructs used for synchronization and communication may however result in calls to the operating system by the runtime system. These are implementation-dependent issues and will vary among runtime systems. For instance, different runtime systems may support different degrees of multiprocessing or, on a single processor, different degrees of multiprogramming. The runtime system uses internal variables to maintain information regarding the status of tasks blocked waiting for rendezvous. The rendezvous management function may interface with the interrupt management function if interrupt rendezvous is supported. The task termination function implements the semantics of the various rules for task completion.

The runtime system provides support for the implementation of all I/O capabilities in the *I/O Management* function. Predefined I/O packages, including but not limited to those described in [ANSI/MIL-STD-1815A, Chapter 14], may be considered to be directly implemented as part of the Ada runtime system or as part of an Ada application program and supported by the runtime system.

c. Binding Management

Ada binding management includes those functions that maintain and check the names and addresses of the objects being used by an application. Several features of the Ada pro-

gramming language such as separate compilation, overloading, and renaming make these otherwise simple housekeeping functions more complex and also more critical.

The functions provided for several features such as representation types, length clauses, and interrupts are implementation dependent and are included in the Ada RTS. Predefined functions provided by the RTS to handle exceptions are invoked at runtime to indicate problems with binding management tasks.

At run time the binding management functions check order dependencies, check that generics are elaborated before they are instantiated, check that subprogram bodies are elaborated before they are called, and that tasks are activated before they are called. Failure of any one of these or other similar checks results in a PROGRAM_ERROR exception being raised by the RTS. The binding management functions also perform runtime checks for accesses to deallocated objects, assignments to shared variables, multiple address clauses for overlaid entities, and use of undefined variables. Failure of any of these checks will result in an PROGRAM_ERROR exception being raised by the RTS.

E.3.3 Views of an Operational Environment

The runtime system is “the set of library functions that is supplied with an Ada Program Support Environment (APSE) that supports the execution of Ada programs on the intended target” [Freedman 1985, 157]. The runtime system, which is provided along with an Ada compiler, is tailored to a specific hardware platform. The runtime system is used to generate a complete mapping between the desired behavior expressed in a high level language application and the machine instructions which will execute on the hardware. Since the capabilities of each hardware target is potentially different, the Ada runtime system associated with the APSE on each target will be unique. Even for functions common between different hardware platforms there is no requirement that runtime system developers provide the support for the Ada application in the same way.

Ada was developed to support embedded systems development. In embedded systems the size of the software is typically an important constraint. Therefore, the Ada language requirements state that applications in Ada shall not depend on the presence of an executive or operating system. For this reason, Ada runtime systems were developed and can provide all the support required by an application. Figure E-5 [ARTWG 1987a, 12] illustrates the view of an operational environment for an Ada application on a bare machine receiving all

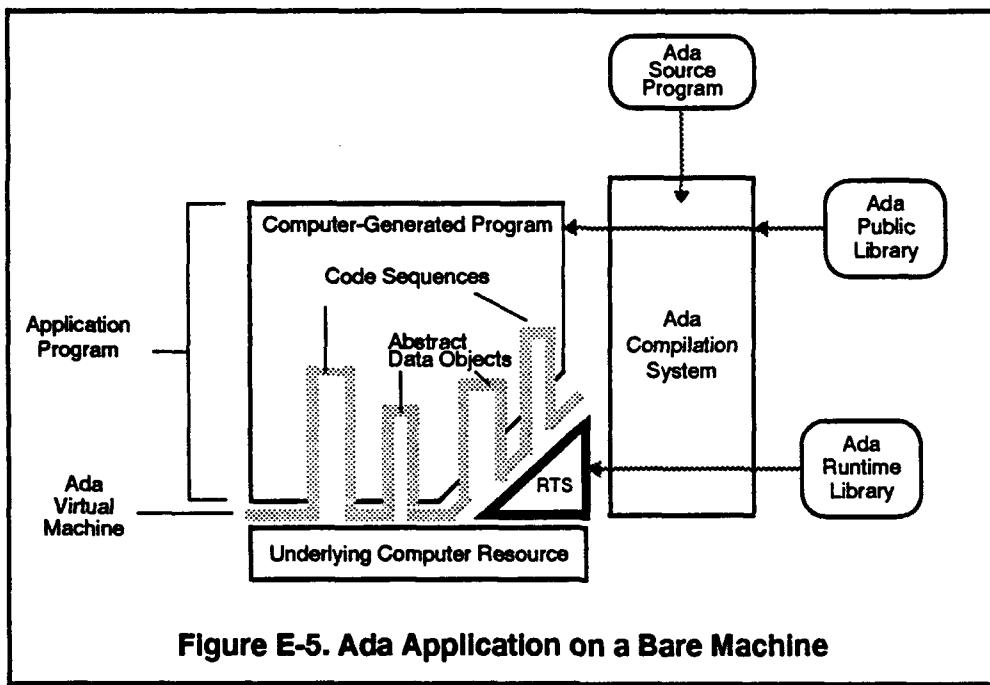


Figure E-5. Ada Application on a Bare Machine

its run time support from an Ada RTS. In this view the virtual machine is created by the compilation system selecting those predefined routines from the Ada Runtime Library needed by the application. The runtime system generated is thus specific to the application.

An Ada application can execute on a bare machine with all the runtime support provided by the Ada RTS. Alternatively, an application can execute on a machine with an operating system or executive. While all support for the executing application can come from the Ada RTS, some operating systems may offer similar services. There exists a choice as to where the Ada application will derive its support. Since the Ada runtime systems are acknowledged to be highly variable in their implementations of the support functions, use of some operating system services may provide the developer with greater control.

Figure E-6 [ARTWG 1987a, 13] illustrates the view of an operational environment for an Ada application receiving its runtime support jointly from the Ada RTS and the operating system. The degree to which the application relies on the RTS determines how large the wedge is. An Ada implementation may use services provided by the operating system, thus decreasing the need to rely on routines from the Ada Runtime Library. Similarly, some features of the operating system may be inconsistent with or unnecessary for the particular implementation of the runtime system and will never be used by the runtime environment.

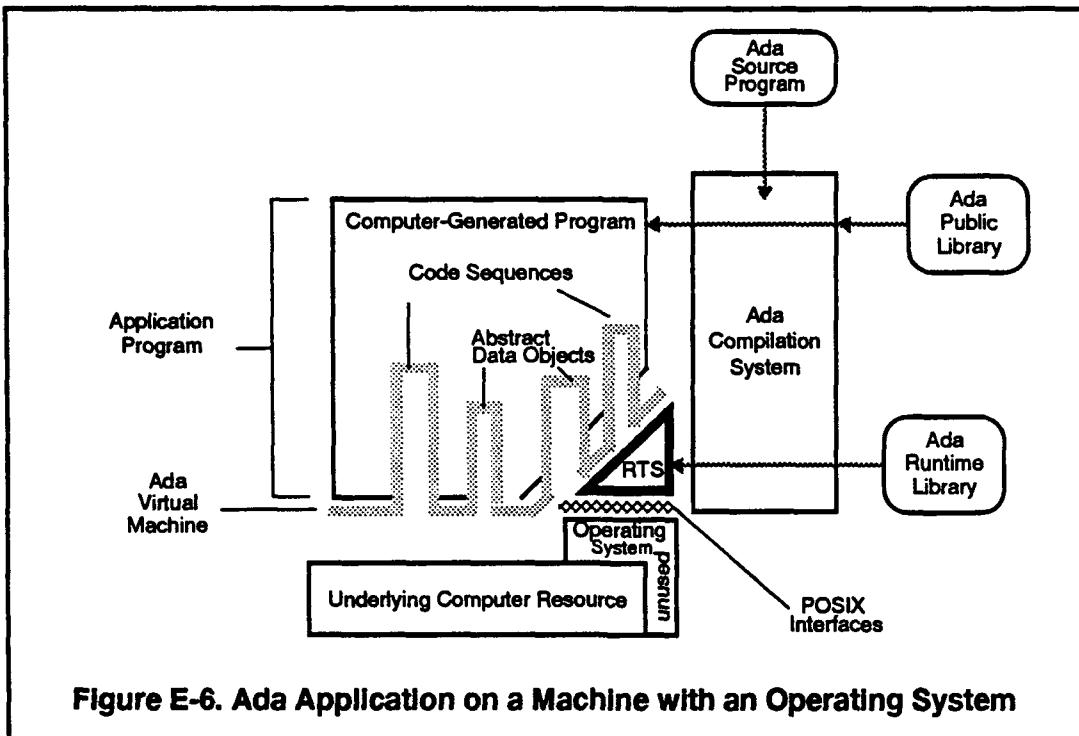


Figure E-6. Ada Application on a Machine with an Operating System

A description of the implementation dependencies permitted by the Ada language standard, entitled *Catalogue of Ada Runtime Implementation Dependencies* was produced by LabTek in 1988 for the U.S. Army Headquarters, Center for Software Engineering. By identifying the features of the Ada language that vary among implementations, software developers can be aware of the potential for their code to be nonportable. In addition, because some implementations of these features may provide better performance than others, the catalogue allows developers to assess the features of a compiler product of particular importance to the application during compiler procurement.

An Ada application developer must be aware of the implementation dependencies permitted by the Ada language standard in producing a product to meet the requirements, whether they are portability- or performance-based. By examining various compiler products and their associated RTSs, a developer can select the compiler that provides implementations he or she requires. Operating systems vary widely in the services provided. Even among the various UNIX and UNIX-like operating systems, the variety of systems calls provided at the program interface hinders application portability. POSIX is a set of standard interfaces based on UNIX system calls. The relationship of the interfaces to the application and the operating system is shown in Figure E-7 [ATRWG 1987a]. The goal of POSIX is

to provide greater application code portability among operating systems with a UNIX-like program interface. POSIX is an IEEE Standard (IEEE standard 1003.1-1988). A more detailed discussion of POSIX is found in Appendix D, Operating Systems.

There is currently an effort to define a standard POSIX Ada binding. Although still a draft standard, the POSIX Ada Binding V4.0 describes a set of Ada specifications to the services provided in POSIX. The binding will allow Ada applications to use the services of POSIX while retaining complete source code portability. However, the draft binding standard does suggest several changes to current Ada compilers and does not address interfaces to the Ada RTS. Interfaces to the Ada RTS are described in *A Catalog of Interface Features and Options for the Ada Runtime Environment* [ARTWG 1987b].

E.4 ARCHITECTURAL ISSUES

The resolution of several programming services issues will influence the overall architecture of the target CCIS. These programming services issues should not be resolved in isolation, but rather must be considered in association with the issues raised in other service areas such as operating systems, networks, and security.

E.4.1 Single vs. Dual Environment Implementation

One significant issue for both the programming services and security services areas is whether the CCIS will provide dual environments, one for software development, and one for software operation, or a single environment allowing both development and operation. The requirements for the WAM specifically state that development of site-specific software will be occurring during the life of the system. The WAM requirements also state the need for multilevel security (MLS). Depending upon how these requirements are refined with respect to policy and architectural changes, these two requirements may conflict. Resolution will require policy, procedural, and/or architectural trade-offs.

Several possibilities for the resolution of this potential conflict exist. Dual environments may be implemented either physically or logically. In the case of the development of software on physically isolated resources, a software developer would log onto a separate computer or network of computers and create software for the CCIS. This software would then be validated and verified before it is installed in the physically separate operational

environment. Alternatively, in the case of the development of software on logically isolated resources, the developer would log onto the same network of computers being used by the operational CCIS software, but would be logically isolated from that software. In this case, special procedure-management software within the operating system would be used to ensure that software being developed or maintained cannot be used in the operational CCIS before it is verified and validated.

A single environment for software development and operation is another approach to resolving the issue. An arbitrary CCIS user may be permitted to write software for his or her personal use on the CCIS, but would not be permitted to share this software across the CCIS without formal approval. Since both the software development and operational environments are potentially available in a single environment, software security mechanisms would be necessary to prevent certain users or developers from performing certain tasks. Trusted software developers would be responsible for generating software for use among the sites on the CCIS.

The decisions on how to implement the environments provided by the programming services area affect the architecture of the CCIS. Dual environments would necessitate the use of simulators or a copy of the operational environment to ensure the software being developed functions as planned. Security concerns may be satisfied more readily however. The use of a single environment would require a more complex security solution. The resolution of the issue of CCIS security for software development and use will include physical, hardware, software, policy, and procedural elements.

E.4.2 Environment Support

A second architectural issue raised in the programming services area is that of the support environment (includes development and operational environments). The CCIS will be a heterogeneous network of computing resources allowing each site to select the hardware appropriate to fulfill its needs. Sites may use personal computers, workstations, graphics terminals, powerful servers, or other hardware to perform their tasks. These hardware platforms may provide a variety of operating system interfaces. The question is whether an integrated support environment is possible and needed at each hardware platform, and if so, should the support environments be identical.

An open support environment will enhance information sharing among tools and people, conformance to processes, policies and standards, tool support, and environment extensibility and tailoring over time. A goal of open support environments is to enable interoperability among heterogeneous platforms. The tools, policies, and software provided as part of an environment are based upon the tasks performed by the site and the hardware platforms used, and may be unique to that site. That is, the support environment at a given site is tailored to the needs of that site.

As part of the support environment tailoring, different sites may support different tools such as programming language compilers or cross-compilers. Not all hardware vendors provide compilers to the variety of languages supported by the CCIS during the transition to the target CCIS. Further, not all hardware platforms will be capable of executing all the programming languages supported across the entire CCIS. However, the multilingual nature of the CCIS is important as it allows the use of COTS and existing software. During the transition to the target CCIS, the tailored support environment may contain programming language translators to ease the migration to modern programming languages and development methods.

A tailored support environment allows software libraries to be distributed to only the sites where they are needed. For configuration management this minimizes the number of updates required when changes occur.

E.4.3 POSIX Services for Ada

A third issue raised in the programming services area is the use of the Ada RTS versus the operating system services. An Ada POSIX binding allows direct access to operating system services. Provided that the operating system services include an Ada POSIX binding, an application developer may write operating system calls within the application code rather than allow the RTS to implement the required functions. This would allow a procurement officer to select an Ada compiler together with a minimal RTS for the support environment; that is, one which relies on the operating system for the implementation of many of the functions provided in the RTS. More discussion on this is in Appendix D, Operating Systems Service. This may also have security implications; see Appendix F, Security Service.

E.5 SUMMARY

A combination of languages and bindings will be required for the future CCIS. Ada has been mandated for use within DoD and is on its way to being Congressionally mandated for use in all government software. However, Ada alone may not be sufficient for some applications and is inappropriate for other applications. For example, Ada does not readily lend itself to developing artificial intelligence and expert systems (AI/ES) applications. This is not to say that Ada can not be used for development, but rather that specialized languages such as LISP and Prolog assist in rapid prototype development of AI/ES applications. Also, using SQL for database queries is more appropriate than using Ada. Non-Ada COTS products will undoubtedly make it into the future CCIS as well. The Ada 9X project to revise the current standard is underway and there are proposed language changes that, if implemented, would make Ada more usable for some CCIS applications than it currently is. In addition, standards groups continue to pursue the development of bindings from Ada to other languages.

A consensus has not been reached on a single, standard tool interface. Common tool interface standards have coordinated, heavily supported efforts underway. The U.S. DoD CAIS-A program and the European Computer Manufacturer's Association (ECMA) PCTE/PCTE+ program are the two largest efforts. While some groups are leaning toward one or the other, the general consensus is that there is no consensus. However, the PCIS convergence project may result in the best of both CAIS-A and PCTE+ and shift consensus to the jointly developed interface.

General software reuse is still an open question. A library of reusable components is a highly desirable item for all large software systems. However, the state of the art in software component reuse is such that technical issues regarding component classification and retrieval mechanisms still remain.

Separating the development and operational environments for the future CCIS is an unresolved security issue. The structure for CCIS programming services is divided into development and operational environments. These two environments represent distinctly different functional groupings that do not overlap and should therefore be kept separate. There are two general ways to keep these functions separate:

- a. Have a single environment representing both the development and operational, with no physical or logical barriers, and which relies upon the integrity of the users and their adherence to security procedures to keep functions separate.
- b. Have a true dual environment with physical or logical barriers to keep functions separate.

It is not clear that one way is more advantageous than the other. In a single environment, development and operational functions would co-exist and as long as proper procedures were followed, a security compromise is not likely. However, inadvertent or deliberate actions could easily compromise the system. On the other hand, a dual environment would provide hardware and/or software controls which would reduce the potential for compromise. A dual system though will be more complicated and is likely to be more expensive to implement.

The future CCIS will benefit from an integrated support environment. An open integrated support environment within the CCIS would provide appropriate enforcement of policies and procedures, more effective use of information, and greater flexibility. The use of an open integrated support environment would lead to the production of better and more cost effective software.

A CCIS support environment would change over time to include different development tools and software libraries as the CCIS evolves toward the target architecture. In addition, each site would continue to tailor their environment to suit their needs. This tailoring may include adding site-specific tools and software, or eliminating unnecessary tools and software.

Within the software development community significant progress has been made in the effort to specify an appropriate framework for the integration of tools and processes. Within the next five years such a framework should be available.

The Ada POSIX binding, which is under development, would provide CCIS software developers with greater control over the behavior of the software they develop. Since there is no effort underway to standardize the many implementations of the Ada RTS, reliance on the operating system services provided through POSIX may be the best approach.

APPENDIX F —SECURITY SERVICES

Security services are those system facilities (hardware and software) that provide protection of classified data within a computer system and in network communications among computer systems. These services are incorporated at various component interfaces (e.g., user, application, operating system, network, database, program development) throughout a command and control system. These services ensure that data passed across the interfaces remain protected. Thus, the issue of security spans most aspects of architectural planning for a generic CCIS. It is due to this structural spanning that security services are referred to as a security architecture.

By law, classified information to be processed within an automated system must be protected at the information's level of classification to prevent compromise. This protection can be accomplished procedurally (e.g., by maintaining the environment and operating the system entirely at the highest level of classification of information within the system - "system high"), or technically (e.g., by employing mechanisms which will enable protection of individual units of information independently at their individual levels of classification - "multilevel"). Since information also may require protection at the discretion of the user initiating the information, additional or extended mechanisms must be provided that enable this discretionary protection.

The bearing of security on generic architectural alternatives depends primarily on the degree of trust required in technical aspects of multilevel protection for shared resources within the system. "Trust" in this context reflects the confidence, founded on a set of accumulated assurances, that the mechanisms employed to implement a policy or set of policies will, in fact, preserve through their actions, the intent of that policy set. Higher levels of trust require that more constraining products be incorporated as components of the CCIS. Additionally, contrasted to the generic CCIS architecture, specific CCIS instantiations must take greater account of the physical and operational environment that place differing and unique requirements on the degree of trust that the array of technical protection must be provide. Security also depends on the availability of component products to provide multi-level protection and meet the required level of trust. Multilevel protection technology in

computer systems is still relatively new and is developing and evolving at a slow pace. Evaluated commercial products are not in widespread use and demand is not strong. Currently there are only multilevel secure operating systems. Much of the research for operating systems products was done in the early 1970's and is only now finding its way into viable products. A new surge of research began in the early 1980's on networking and in the mid 1980's on DBMSs. Much of this later work remains as paper studies and worked examples to serve as proof of concepts are only now beginning to appear.

Although multilevel security is part of the given CCIS requirement, it is unlikely that standards for validated, trusted multilevel security will be in place in time for a 1995-97 development. DISA, the task sponsor, is actively working the security issue. In July of 1990, the Director, DCA, established a joint DCA-NSA "Defense-wide Information Systems Security Program." Pending the report on this activity, no recommendations for security beyond system-high mechanisms are made in this target profile. Rather, a separate working paper on security is in preparation. It is in part tutorial and in part an issue paper addressing the large number of yet-to-be-resolved areas of multilevel security. The white paper on security services establishes the security issues to be addressed in developing the architectures for future Defense CCISs and describes the generic CCIS security architecture to be used as an acquisition target.

APPENDIX G — USER INTERFACE SERVICES

G.1 INTRODUCTION

User interface services control how applications appear to the user and how the dialogue between the user and application performs. Graphics capabilities, which can have a critical impact on CCIS portability and interoperability, are treated as part of user interface services. This appendix describes the types of human-computer interaction that will be supported by these services and outlines the user interface architecture. This architecture is based on the proposed NIST Application Portability Profile [NIST APP/OSE 1991]. Although input/output devices and the hardware needed to accommodate CCIS graphics processing are not part of the architecture itself, these are potentially critical concerns. For example, the mobility necessary to CCIS survivability will be directly affected by the size and durability of particular device. Human factor concerns similarly are not part of the architecture, but must be addressed. Various special user interface technologies, whose use should be considered for the 1995 CCIS, are discussed. These include scientific visualization, hypertext, desktop and teleconferencing, and desktop video.

G.2 HUMAN-COMPUTER INTERACTION

Communication style defines the ways in which operations are represented and performed. Consistency of behavior is an important interface characteristic. The basic style is a dialogue where the user types instructions to the computer and receives text responses. Direct manipulation allows the user to manipulate a task representation directly, using, for example, iconic representations of objects and operations. Group interfaces allow users to work cooperatively and may allow users to interact through a variety of data media. Important characteristics of communication styles are consistency, ease of learning and use, and feedback. The CCIS user community includes people who will use a variety of applications and a standard "look and feel" will help ensure a uniform interface throughout the commu-

nity and across applications. Ease of learning and use depends on two types of memory: *recognition* memory that helps a user connect displayed command alternatives with a given action, and *recall* memory that helps a user remember the command for a desired action. Adequate feedback requires that a user, entering a command, rapidly receives some indication that the command is being processed, although care must be taken not to disrupt the user's concentration. These and other aspects of communication styles are discussed in more detail below.

G.2.1 Basic Interaction Types

The basic interaction types include command languages, menus, and form filling. Command input may be either application or user directed. A command language provides a relatively free form of input. Little or no prompting is given and the user independently selects the action to be performed at each step. Providing the greatest flexibility, this mode of interaction is suited to the skilled and frequent user, usually permitting fast task completion for the experienced user. Macro definition facilities can allow a user to define new commands and thus to personalize the environment. They also permit extensions designers did not foresee or that are useful to a small part of the user community. A macro facility may be a simple template capability or a full programming language with arguments, conditions, numeric and character types, and screen manipulation primitives, plus, perhaps, library and editing tools.

Design of a command language should be based on a user's conceptual model of the system and its input devices. It requires an understanding of the user community. A natural structure is important. It will help users encode their semantic knowledge of the system. Cognitive insights can be applied in the grouping of information within a command to aid learning and command production. The relative merits of positional and keyword syntax for novice and expert users can be exploited.

Menus provide a simplified interaction style that reduces the possibility of keying errors. The explicit list of options in menu-based dialogue eliminates the need to memorize command names. It provides a clear indication of permissible actions, structuring a task to guide the novice and infrequent user. On the other hand, menu selection is often considered slow and clumsy by skilled users. Waiting for lengthy menus to be displayed can be annoying. Research has addressed many aspects of menu design, including names *vs.* numbers

vs. icons as menu choices, semantic organization of items in a menu, and alphabetical vs. probability-of-selection vs. random vs. positionally constant menu arrangements. Major design decisions will concern menu system structure, menu position, method of menu invocation, menu traversal, item selection, and graphic layout. These are in turn affected by available screen space, understanding of human cognitive processes, and the requirements of a particular application. Approaches for increasing speed for expert users include allowing typeahead for known menu choices, assigning names to menus to allow direct access, and the creation of menu macros that assign unique names to frequently used combinations of menu selections.

In text menu control, menu selections are made by typing the number of a menu entry, typing the first letter of an item, or using arrow keys or a pointing device to move among menu entries. Default selections can be indicated using techniques such as reverse video, blinking, and brightening, or a separate "current selection" field. In graphical menu control, symbols, icons, color, or patterns are used instead of textual descriptions with a graphic input device for pointing and selecting. Care is needed to ensure that graphics items are easily selectable by the available input devices. Icons can be used to attempt to convey the meaning of an action. The major goals of a window management system for the CCIS user interface include high-performance with high-quality text and graphics; network transparency; configurability and extensibility of the window management system; portability and device-independence for both applications and the window management system. It should be capable of supporting many different applications and management interfaces with true multitasking of applications. The increased use of window management systems in networked environments has encouraged standardization of the underlying architecture of window management systems.

When entry of many fields of data is required, for example, in questionnaires, spreadsheets, or other tabular display applications, form filling is an appropriate input style, allowing visibility of the full complement of information and giving users a feeling of being in control. Few instructions are necessary since this approach resembles familiar paper forms, although knowledge of special keyboard functions will probably be required. Design considerations include dealing with multi-screen forms, mixing menus with forms, use of pointing devices, use of color, handling of special cases, and integration of a word processor to allow remarks.

G.2.2 Direct Manipulation

Direct manipulation interfaces are event-driven rather than command-driven and inherently object oriented. Physical actions such as “pushing” labelled button images on the screen replace menus and commands. A simple example of a direct manipulation interface is the use of arrow keys to direct a cursor to a specific point on a display instead of using commands that would require input of numerical coordinates. The key characteristics are continuous representation of the object of interest with rapid, incremental, reversible operations whose impact on the object of interest is immediately visible. Direct manipulation interfaces typically use elaborate graphics, alternative ways to achieve a given effect, mode-free operation (the user can give any command at virtually any time), and rapid semantic feedback. Because the contribution of user actions to goals is clear, such interfaces can help novices quickly to learn by observation and allow for continual development of skill without special training. They can support layered learning that allows immediate use by a novice as well as highly efficient use by experts, with users continuing to use the same concepts as they gain experience. There is a lack of documented design strategies that guarantee the development of easy to use, or easy to learn, interfaces, consequently, prototyping and field trials are particularly important in the development of direct manipulation interfaces.

Windows are a basic mechanisms for direct manipulation and for many other aspects of the modern human-computer interface. A window is an area on a computer display, usually rectangular and usually delimited by a border, that encapsulates all or part of an application or function. It often contains a particular view of some data. Pull-down menus, dialog boxes, and message boxes used to separate specific segments of the user-system dialog from the main application, can be considered windows. It is often stated that windows can be used to represent the way people actually work at their desks, providing a useful way of dividing the display according to logical structure. This allows use of spatial mapping to help structure work, allowing transfer between tasks by simply transferring attention from one window to another, while providing place holders and a visible memory. This so-called *desktop metaphor* can be enhanced through the methods used for communicating window operation requests to, and receiving feedback from, the system, in particular, by treating windows as objects that can be accessed and modified by users.

The goal of three-dimensional (3D) user interfaces is to provide a flexible and natural dialogue in inherent 3D environments. Stereoscopic computer graphics can be applied to

both display devices and video. Apparent depth perception can provide extra clues to the true shapes of objects, making it easier for users to manipulate some forms of visual information and improve the speed of comprehension. Studies have shown that, regardless of input devices, test subjects are nearly twice as precise when they use stereoscopic display as opposed to a flat display. Virtual image displays can be used for interfaces where the user appears to be in a 3D environment. The entire field of vision is filled with computer generated images and multidimensional input devices interact directly with the underlying application. There are typically three types of movement: (1) the user may move independently, (2) gesture-controlled *flying* signalled by a user's pointed forefinger, and (3) reaching out with a hand whose image intersects with a displayed object to allow grasping and moving that object. The basic ideas for virtual image displays were introduced in 1983 but, though the necessary hardware and software is commercially available, several problems remain. In particular, better means for devices that can produce sound and tactile feedback. Even so, the ability to give a user the feeling of directly interacting with the application holds great opportunities for many military applications, for example, various kinds of interactive training, exploring and interacting with 3D environments such as buildings and landscapes, or supporting remote surveillance and telerobotic planning.

G.2.3 Group Interfaces

Most user interface research focuses on single-user systems. Computer Supported Cooperative Work (CSCW) addresses services for people who work in groups. Its products, referred to as *groupware*, assist groups in communication, in collaboration, and in coordinating their activities. In many respects, CSCW moves beyond human-computer interaction to address computer-enhanced human-human interaction. It embodies at least five different disciplines: distributed systems, communications, human-computer interaction, artificial intelligence, and social theory. Several modes of worker collaboration based on the concept of shared workspace have been recognized [Ishii 1990, 19].

Group interfaces must consider such factors as group dynamics and organizational structure. In particular, group interfaces differ from single user interface in that they depict group activity and are controlled by multiple users. Additionally the higher level of activity and greater degree of concurrency tends to increase interface complexity aggravating some existing problems. For example, management of already limited screen space is even more difficult when each user can create windows that appear on other users' screens. In this

case, one approach being investigated is to group windows into *rooms* corresponding to particular tasks and allow participants to move from room to room or be *teleported* by other users. Other issues range from handling group focus and limiting distraction to the lack of group interface toolkits.

Several approaches are used for integrating communication and information processing technology. At one end of the range are computer-based techniques for handling information stored in computers, such as shared-window systems and special purpose multi-user applications running on network workstations. At the other end are telecommunication-based techniques for handling information external to computers, such as techniques for sharing drawing surfaces. A mid-range, desk-based approach is to use video overlays for fusion of the computer screen and desktop images to allow the sharing of information contained in printed material and computer files with real time information such as hand gestures and hand-written comments. Most systems assume some intentional action to initiate communications between parties, but approaches for non-interactive communications, for example, electronic bulletin boards also exist. Two applications of group interfaces, teleconferencing and desktop conferencing are discussed in Section G.7.

G.3 USER INTERFACE ARCHITECTURE

The user interface is modeled on the NIST Application Portability Profile (APP) open systems environment (OSE) profile that is defined in terms of open system specifications organized into major service categories. In the case of user interface services, the goal is a clean separation between the user interface and other parts of the system to allow reusing a given application with multiple interfaces, ensuring consistency among a family of applications using a common interface and providing independent tools for application and interface developers. The user interface service specifications are based on the NIST User Interface Services Reference Model (UISRM) [Kuhn 1990]. This is a conceptual model defined in terms of layers; specifically, below the application are Dialogue, Presentation, Toolkit Components, Subroutine Foundation, Data Stream Interface, and Data Stream Encoding layers. The layers are arranged from bottom to top in ascending complexity. The bottom layers contain primitive constructs on which upper layer functions are built.

G.3.1 Dialogue Layer

The Dialogue layer mediates the user's interaction with applications. It is implemented by a User Interface Management System (UIMS) where the interaction is specified by a mapping that associates user actions with application actions. Terminal Management (TM) is another aspect of dialogue control that addresses presenting data from several sources on a single display and supporting multiple users and displays attached to one application.

A UIMS handles all visible parts of a display and all aspects of dialogue between the user and application, to achieve two primary goals: to separate completely the user interface code from the application code to provide a standard "look and feel," and to support more abstract specifications of the user interface that describe the appearance of the interface and the kinds of interaction it supports. Different types of notations are available for particular kinds of interfaces, for example, graphical notations that can be used to partially define the interface by placing objects such as menus and buttons on the screen with a mouse. Object-oriented notations are an important new class that provide a computational link between the input and output that the application can modify to provide semantic processing. Consequently, these are suitable for highly interactive, direct manipulation interfaces. Typically, a UIMS interprets the user interface specification at runtime, requiring the application to handle such things as screens, aborting, and prompting and employing a window management system to carry out the necessary display and event handling. By helping developers to combine and sequence interaction techniques, and to manage functions like feedback loops, error handling, checkpointing and restarting, a UIMS can enable an iterative development approach for (semi)automatic construction of graphic-based user interfaces, allowing alternative designs to be cost-effectively prototyped. The IEEE project 1201 (P1201) is looking at UIMS standards but, as yet, there are no current international standards that can be exploited for the OSE profile. Examples are OpenLook and Motif.

OpenLook, jointly developed by Sun Microsystems and AT&T, is a device-independent UIMS that supports multitasking graphical user interfaces. The object oriented user model is intended to support a wide range of users. For example, a help facility is provided to support novice users, window properties provide consistency for casual users, and keyboard accelerators support expert users. The Desktop File Manager lets users view files and directories in iconic form and perform operations by selecting and dragging icons rather than remembering commands, thus extending the direct manipulation desktop metaphor used in many window management systems. *Pushpins* enable users to "pin" menus and pop-up

windows to the screen for repeated use. The base window management system, a combination of X-Windows and NeWS (Network extensible Window System), provides access to graphics resources over a network and compatibility with other X implementations and PostScript language systems. The OpenLook user interface is independent of particular toolkit implementations and several network transparent toolkits have been developed.

Motif was developed by the OSF (Open Software Foundation). It is a hardware, network, and operating system independent product that incorporates Digital Equipment Corporation's X toolkit and user interface language for describing visual aspects of the user interface, Microsoft's Presentation Manager-style behavior for user skills portability, and Hewlett-Packard's 3D appearance. Motif conforms with the X/Open Consortium's Portability Guide Issues 3 Standards for Native Language Support, providing support for both 16-bit and compound strings for localization in Asian and European languages. Motif also provides an object-oriented User Interface Language for specifying the user interface appearance. See the Presentation layer discussed in Section G.3.2.

Future UIMS development will address expanded user abstractions, improve image generation, and support the ability to maintain user profiles and tailor the presentation of an interface to a user's preference. Isolated examples of these features are already available. Interactive design tools, sometimes called fourth generation UIMS, are another trend in UIMS development. These are intended to promote the productive use of a toolkit or UIMS via direct manipulation of the interface implementation, instead of by means of a library or language. They are addressing the issues of dynamic interface objects found in visualization interfaces (see Section G.7.1) and starting to exploit artificial intelligence (AI) techniques for cognitive interfaces (see Section G.9.3).

TM provides for handling the same data at different levels of abstraction. A graphics image, for example may need to be manipulated as a display list, as a geometric object, or at the bitmap level. TM controls how the logical structure of a dialogue is mapped onto resources, an area addressed in ISO 10184, a standard for support of multi-function workstations. This standard permits the establishment of a general network of processes with dialogues between them. The dialogues may be of a variety of types, such as virtual terminal or bitmap graphics. The standard does not itself define the operation of an individual process, nor does it define the data stream for a particular dialogue type (these are specified by other standards). Where a process has input parameters that may be adjusted, such as the positions and priorities of the various windows in a window system, these are provided.

ISO 10184 is related to Document Transfer and Manipulation, CCITT user interface standards (SC18), Forms Interface Management System (FIMS, JTC1 SC22), and window management (SC24).

G.3.2 Presentation Layer

The Presentation layer determines the appearance of the user interface, including such features as size, style, and color. The appearance specifies how the components provided in the toolkit layer should be composed to create windows. Control of interaction style is achieved by isolating the implementation of interaction techniques (provided as code libraries of common interface components) to provide a consistent style across several user interfaces. The component libraries generally use a fixed style and must be edited to change that style. They provide high-level programming abstractions for building user interfaces, for example, abstractions that allow the creation of windows, menus, and scrollbars.

There are two approaches to structuring these libraries. The first employs a simple collection of procedures that can be called by an application. The second type uses an object oriented programming style to enforce a separation of *subjects* (abstract objects) and *views* (interactive objects). This approach is exemplified by DEC's X Toolkit which allows composition of primitive interactive objects, such as menus, based on specific composition schemes such as tiling, or overlapping. Layout of composite objects is achieved by a geometry manager that can be replaced at run time to change the layout strategy and present different, independently customizable interfaces to the same data. Another example is provided by the Andrew Toolkit, developed by the Information Technology Center, a joint endeavor between Carnegie Mellon University and IBM Corporation [Waldo 1991]. This toolkit supports building applications using software packages, called *multimedia insets*, designed for the display and modification of various types of information. Insets are currently available for text, raster, table, animation, hyperlink, and calendar types. Insets supporting sound and voice are under development.

NIST has selected XVT (eXtensible Virtual Toolkit) for its APP. XVT has also been selected by the IEEE P1201.1 Standards Committee as the basis for a standard application program interface for portable graphical user interface applications. Based on X, the XVT toolkit provides an example of a look and feel independent approach being developed to overcome the limitation of existing UIMS to restricted ranges of interfaces. It is intended

to allow users to choose their own UIMS, providing for evolution in UIMS technology and enabling easier porting of applications to proprietary systems. XVT libraries form a common programming interface with implementations for several window systems, including Motif. An application is developed with XVT library calls and then the appropriate library for the desired UIMS is used. To resolve differences in handling, say, font structures and the relationships between windows, XVT implements its own comprehensive virtual UIMS toolkit. It comes with a resource compiler that implements a portable resource specification language and an object oriented, MacDraw-like program written using XVT. With its support from NIST and IEEE, XVT is recommended for the OSE profile.

G.3.3 Toolkit Components and Subroutine Foundation Layers

The Toolkit Components provides components such as menus, push-buttons, scroll bars, or help boxes for building an application interface. While these components vary with vendors, they typically contain most of the common user interface elements. IEEE P1201.1 is addressing this layer. The Subroutine Foundation uses features of the Data Stream Interface to provide the means to build window interface components and provides functions such as initialization and destruction of objects, management of events and object hierarchy, and the saving and restoring of the interface state. It is implemented by a window management system.

X-Windows offers the advantage of industry standardization for the OSE profile and the specification for the Toolkit Components and Subroutine Foundation layers is provided by FIPS-158, X Window System. Often called simply 'X'. X defines a C source code level interface to a network-based bitmapped graphic system. Developed at MIT, X is a standard part of AT&T UNIX System V (Release 3.2 and later) and part of OSF the operating system. Like most window management systems, it allows devices to be shared among several processes at the same time. Unlike other systems, it allows access to graphics devices from remote sites. X is undergoing further development. The graphics model is being extended, for example, to include 3D graphics support. Also, Display PostScript is being included for support of two-dimension (2D) graphics. NIST will provide third party conformance testing services through the National Voluntary Library Accreditation Program when test suites and testing policy for FIPS-158 become available.

In addition to IEEE P1201, X is the subject of ANSI standards projects. ANSI Committee X3H3.6 is considering options for efficient OSI-compatible support for X. Other ANSI committees with an interest in window management are X3H5 (Virtual Terminal and User Interface), X3T5 (OSI), and X3V1 (Documents). The Inter-Client Communications Conventions Manual (ICCCM) from the MIT X Consortium is also recommended for the OSE profile. The intent of ICCCm is to enable applications to share data with other compliant applications, share network resources, and to support interoperability of applications and resources running in networked, open system environments. FIPS 158 will contain the ICCCm and the X Consortium is to define how applications will communicate with one another.

The use of windows imposes some hardware requirements. A large screen size is needed so that users do not need to spend a disproportionate amount of time moving, resizing, and scrolling windows. Slow processing speeds and limited memory can cause window operations to strain computing resources, limiting effective feedback, the use of animation, and the behavior of dynamic interfaces.

G.3.4 Data Stream Interface Layer

The Data Stream Interface specifies a function call interface to build messages defined in the Data Stream Encoding layer. In this manner it supports network communication and provides access to basic graphics functions from Layer 0. It may also support system functions such as error handling and synchronization. The APP specification comes from FIPS 158, Xlib - C Language X Interface. IEEE Project P1201.4 on the X Library is working on this area.

G.3.5 Data Stream Encoding Layer

The Data Stream Encoding layer defines the format and sequencing of byte streams passed between client processes that require services and server processes on the same or separate platform that provide the services required. As a specification of message formats, it is independent of operating system, programming, language, or network communications. The APP specification is provided by the X Window System Protocol, X Version 11 from FIPS 158.

G.4 GRAPHICS SERVICES

Graphics subroutine libraries are responsible for graphical information sent to and received from the screen and associated input devices. They provide primitive graphics functions such as "draw line" and "draw circle." Several standards that provide a textual specification for graphics functions have been developed and must be supported by both an implementation of the commands and language bindings that provide the ability to access the standards from different programming languages. In order to promote user interface separation, calls to a graphics package should be incorporated into toolkits or UIMSSs, not embedded in application code. The primary standards for graphics subroutine libraries are discussed below.

Independence of particular I/O devices can be achieved by requiring an application to interface with a graphics package that produces device-independent output. This output is processed by a software module that generates device-dependent commands to control the actual device as well as providing additional functions such as controlling the appearance of graphical primitives which allow powerful functionality at a high level of abstraction, without sacrificing system performance. Graphics metafiles are a common mechanism for supporting graphics device independence. These are formatted disk or magnetic tape files, created by a generator in association with a graphics package, and contain graphics commands and data. They are used to store or transmit pictures among diverse applications or across separate programming environments. Another major use of graphical metafiles is the development of symbol libraries that allow the reuse of graphics symbols in building complex pictures and drawings. Here again, ISO standards will support the OSE profile.

G.4.1 Graphical Kernel System (GKS) and GKS-3D

ISO 7942 GKS provides a functional description of a 2D graphics interface to yield the basic graphics support required to produce computer-generated pictures. It shields the application from hardware particulars by using an abstract description method that presents uniform output primitives, uniform input classes, and uniformity in operations pertaining to output attributes and picture creation, manipulation, and storage. GKS is widely supported and is good for simple interactive applications where portability is more important than functional extensibility and performance. The Department of Defense has funded the design of a graphics package for the WWMCCS Information System (WIS) project that

extends the functionality of the GKS-Ada implementation [Foley 1986]. GKS supports a standardized metafile that provides the capability to store and transmit pictures. This metafile is an audit trail of the GKS commands used to generate a picture at the workstation level. It is suitable for transaction recording which, in turn, aids activities such as restart and recovery. ISO 8651 specifies language bindings for Fortran, Pascal, and Ada. Bindings for C, LISP, and MUMPS are under development.

The GKS FIPS, FIPS 120, was published in April 1986 and is based on ISO 7942 and ANSI X3.124. NIST has licensed a conformance testing suite for GKS and currently operates a GKS Test Service to test implementations against the FIPS standard.

ISO 8805 Graphical Kernel System for Three Dimensions (GKS-3D) extends GKS.

G.4.2 Programmers Hierarchical Interactive Graphics System (PHIGS)

ISO 9592, PHIGS, provides multilevel graphics data structuring, geometry manipulation, and both 2D and 3D graphics. PHIGS has methods for real time modeling and other higher-level programming capabilities and is suitable for interactive and highly dynamic applications such as C², Computer-Aided Design and Computer-Aided Engineering, simulation, and scientific visualization applications. It has a centralized hierarchical data store and support for multiple color models. PHIGS does not, however, provide raster functions; neither does it support window managed environments on workstations. PHIGS extensions for basic surface rendering have been proposed as part of ISO 9592, PHIGS+. The PHIGS language binding for Ada is also defined in ISO 9593, bindings for Pascal and C are under development.

FIPS 153 is based on ANSI X3.144 and X3.1441 as well as the ISO standard. NIST is developing a PHIGS Test Suite, the first version of which is expected to be completed by late 1991.

G.4.2.1 PHIGS Extension for X (PEX)

The PEX graphics subroutine library is a PHIGS extension in the form of a protocol specification and includes much of the functionality of the proposed PHIGS+. The PEX architecture is object oriented for a flexible and extensible environment; its resources include lookup tables, pipeline contexts, renderers, name sets, structures, search contexts,

and fonts. It allows each window on the display to act as an independent virtual 3D graphics workstation, providing support for imaging processing. It supports the several color models, the CIE color standard, and direct color types as a 16-bit integer value. Any of PHIGS, PHIGS+, or GKS-3D programming interface libraries can be built on top of PEX, allowing calls to not only these but also the Xlib and X Toolkit libraries discussed in the previous section.

PEX is intended to overcome the limitation of X to 2D graphics. There are some basic differences between X and PHIGS. For example, PHIGS defines input prompting and echoing as occurring asynchronously with respect to the application, an *internal control* model for user interaction. A typical X application, however, has an event processing loop that monitors and performs the user's input commands, called an *external control* model. This incompatibility can be resolved through use of a tightly coupled interrupt handler (which limits portability) or by a separate input manager process (which complicates I/O coordination) [Sung 1990]. The PEX definition avoids this issue to allow different implementations. Another concern is in the handling of system resources. While X shares resources between all applications, PHIGS does not consider the operating environment and has no notion of sharing. The issue of system resource ownership and external effects on the PHIGS application must be addressed to combine these different approaches.

The X consortium implementation of PEX, the Sample Implementation effort, began in late 1988. Most vendors have some PEX implementation efforts in progress and the first commercial versions are now available.

G.4.3 Device-independent/device-dependent (DI/DD) Graphics Standards

ISO and ANSI have prepared DIS 9636, the Computer Graphics Interface (CGI) standard. It defines a system-level interface to a virtual graphics device to provide a standard specification of efficient control and data exchange between device-independent graphics software and one or more device drivers. Device dependencies are allowed in limited circumstances, such as when dealing with raster entities. Character, binary, and clear-text encodings are provided. Language bindings for FORTRAN and C are under development. CGI is expected to be published in the final quarter of 1991, but early versions implemented by IBM and AT&T have yielded workstation managers that are already becoming the computer graphics industry's de facto standards.

G.4.4 Rendering Standards

Rendering is the process of translating geometrically described objects into displayable pixel-based pictures. It is not addressed by existing 3D graphics standards, nor by any active standards committee. The OSE profile can take advantage of the RenderMan interface, developed by Pixar working with other graphics companies, for device independent 3D rendering. It is a scene description interface that partitions the generation of images into the distinct areas of interactive modeling and noninteractive rendering. It supports:

- a. A definition for all data that can be output as part of a 3D scene description.
- b. Procedural primitives that allow a modeling application linked to a RenderMan renderer to supply an arbitrary refinement subroutine as the description of a procedural primitive, such as a fractal.
- c. The basic set of 3D graphics functions, a hierarchical transformation stack with a full set of transformation operations, orthographic and perspective viewing transformations, and device independent image-size control.
- d. Antialiasing, filtering, and image quantization quality controls to give the user the power to generate high-quality images without annoying artifacts.

The RenderMan Shading Language provides user-extensible control over shading using geometric information instead of the usual mathematical equation that is based on a simple model of the reflection of light. The RenderMan Interface Bytestream protocol provides both binary and ASCII encoding archive formats for a complete RenderMan scene description that is suitable for network database transmission as well as file storage.

G.5 INPUT/OUTPUT DEVICES

G.5.1 Display Devices

There are three types of display devices: CRT, flat panel, and projection. The choice of a particular type for a given computer system depends on display characteristics, such as color, screen size, and resolution and operating characteristics such as mean time to failure, power consumption, memory, and bandwidth requirements. To some extent, screen size is a function of both physical size and display resolution. Even so, large screens are preferable to smaller screens with higher resolution; for example, low resolution characters constrain

the use of visual coding or good graphics design principles. In particular, high resolution bitmapped displays for the creation and manipulation of more detailed representation of characters and images at the pixel level allow more information to be conveyed visually. Environmental factors, such as tolerance for vibration and resistance to surface glare from reflected light, are also relevant. Multiple displays may be required to overcome the space limitations of a single, small workstation display and to provide suitable devices for diverse presentation materials such as text, maps, or photographs. CCIS selections will also consider the rapid development of display technology; for example, the Defense Advanced Research Projects Agency is funding research into these technologies and their applications. In addition to both strategic and tactical C² applications, this research is looking at workstations, avionics and instrumentation, and military vehicle instrumentation.

G.5.1.1 CRT Displays

CRTs represent a stable technology that has been available since the early days of computing. Research into flat tension mask CRTs may, however, impact future systems. These CRTs provide an ability to avoid surface distortions, allow two-sided antireflective coating and a brighter image that is likely, to offer improved user performance.

CRT displays range from 2 to 30-inches. Interactive graphics systems typically employ a special purpose display processor that is used to control the operation of the display device. In a graphics monitor, as opposed to a workstation with graphics capabilities, this might be the only processor. Typical basic tasks for the display processor include display of line segments and character generation. In addition to the standard character set available on all systems, some systems allow user-generated character patterns to be stored and reproduced by the display processor. Advanced display processors may perform a variety of additional functions, such as generating various line styles (dashed, dotted, or solid), displaying color areas, producing curved lines, and performing certain transformations and manipulations on displayed objects. The display processor may refresh the screen or this task can be assigned to an additional processor, called the display controller. The display processor also interfaces with interactive input devices, such as a light pen. Key characteristics include:

- a. Pixel Architecture. Relates to the number, position, and color of adjacent phosphors that are impacted by the electron beam as a single pixel. *Triad* and *quad*

are common pixel architectures. Square pixels are used to aid operations such as rotation, although rectangular pixels are preferable for video.

- b. **Pixel Density.** A measure of each display's ability to discriminate fine detail, possibly specified in terms of pixels per display width unit.
- c. **Aspect Ratio.** The ratio of screen height to width which affects, for example, the sensitivity of the periphery of the human eye to flicker. Commonly 3:4 for computer displays, systems for symmetrical images, such as radar, may use a 1:1 ratio. Standard NTSC cameras produces an image with an aspect ratio of 4:3, proposed HDTV standards uses 16:9, and most motion picture films use 2:1
- d. **Frame Rate.** The frequency at which changes might be made to the image.
- e. **Refresh Rate.** The frequency with which an image is brightened, varying from 30 to 65 screens per second. Computer systems usually perform a *progressive scan* where lines are brightened consecutively, whereas in the *interlace* approach used for video the image is split into two parts with all odd lines redrawn before all even lines. Video screens must be refreshed 50 to 60 times a second for a stable image. NTSC video has a frame (raster lines) of 525 horizontal lines occurring as two fields of 262.5 lines and alternating halves of the display are refreshed every 1/60 second. Proposed frames for HDTV are 1050, 1125, and 1250.
- f. **Coordinate System.** Defines the coordinate origin, typically the lower-left screen corner with screen surface represented as the first quadrant of a 2D coordinate system.
- g. **Frame Buffer.** The number of lines per screen using standards such as Video Graphics Adapter (VGA) and Extended Graphics Adapter (EGA).
- h. **Sync.** The information in a video signal that causes horizontal retrace before starting a new line and vertical retrace before starting a new field.

Pertinent image quality characteristics include:

- a. **High Ambient-Light Readability.** For example, readability in sunlight with preservation of contrast and chrominance.
- b. **Image Stability.** Refers to a blurring of the display resulting when a CRT image is not refreshed in precisely the same place from one frame to the next, the image persistence is too short, or the refresh rate is too low.

- c. Luminance and Contrast. For emissive displays, contrast ratio is defined by ANSI/HFS 100-1988 as a minimum ambient illuminance of 500 lx. Luminance is the luminous flux emitted by or reflected from a display surface.
- d. Polarity. Negative contrast displays are those in which the symbols are darker than the background, sometimes called *positive image* displays.
- e. Resolution. The displayed resolution is defined in terms of overlapping picture elements and their visual separation. The maximum resolution is around 100 lines per inch. The resolution recommended for a given display is a function of a number of variables, including viewing distance, ambient light level, task requirements, and screen content.
- f. Viewing Angle. The maximum angle from the normal (both horizontal and vertical) within which the information displayed continues to be intelligible, is determined as the point at which the contrast ratio reaches a predetermined level, usually in the range of 1.4:1 to 3:1.

These characteristics will be considered with respect to environmental factors such as glare. Sources of glare can be classified as resulting in user *discomfort* or *disability* that interferes with extracting information from the display. Antiglare treatments can be applied to the display surface if glare cannot be removed from the environment.

CRTs may employ either a random or raster scan drawing system. In a random scan system, a structured display file contains a set of commands, for example, line or character drawing commands, which is accessed by the display processor to repeatedly regenerate the image to refresh the screen. In a raster scan system, the refresh storage area, termed the *frame buffer* or *bitmap*, represents an image in terms of intensity information for each screen position. Each position in the bitmap is called a picture element or pixel and pixel positions are organized as a 2D array. The number of pixel positions is the resolution of the display processor, or bitmap. This may differ from the CRT resolution though, ideally, CRT resolution should be equal or greater than that of the display processor.

These two drawing systems possess different advantages. Raster systems serve well as an archival format and can deal directly with images scanned from an external source, but the generation of complex images is relatively slow for structured display files. Bitmaps save recomputing an image for, say, rapid panning and scrolling of a window over a larger display area, but the transfer of images of differing resolution causes some problems, and bitmap size is constrained by physical limitations. Another disadvantage of bitmaps is that

any structure used in constructing an image is lost and, hence, it can be difficult to identify logical components of the image. A graphics system need not employ one approach exclusively and some combination of bitmaps and structured display files is perhaps more appropriate than either alone. They may be combined in several ways, primarily by integrating random and raster editing on separate layers within the same system or adding bitmap primitives to a random system. As well as imposing different performance overheads, each type of combination is suitable for different applications. For example, integration via separate layers supports accurate geometry and dimensioning and quick capture through scanning. Adding bitmap primitives to a vector system provides the flexibility needed for graphic art and supports a resolution-independent display capability. Commercial graphics systems employing both approaches are available.

G.5.1.2 Flat Panel Displays

Flat panel displays are inherently stable because images are generated by fixed electrodes that cannot move in the image plane. It is estimated they can reduce display volume by 50 to 75% over CRTs, while increasing reliability by a factor of 4. Flat panels are expensive, for example, a 19-inch color flat plasma panel monitor costs around \$300,000, and exhibit relatively poor performance with respect to a CRT. Nevertheless, the cost and performance of flat panel displays may rival CRTs by the mid-1990s [Tannas 1989, 34-35] and offer great potential for the C² environment where limited space and the need for mobility have hitherto minimized the usefulness of displays.

These displays use either an emissive or shuttering approach. In the first case, plasma or thin film electroluminescence (TFEL) technology is used to provide pixels that are energized to emit light. TEFL displays are promising where high contrast, high luminescence, and high resolution are critical factors. Although this technology holds promise for the development of full-color TEFL displays compatible with television signals and computer-driven color models, its power requirements limit use in portable battery-driven environments [Synder 1988]. In plasma panels the size of plasma capsules limits resolution. They are usually orange and flicker-free, displaying up to 62 lines of 166 characters. For shuttering, a signal is used to filter a posterior light source, both spectrally and with respect to intensity. LCDs are the most widely used and 5-inch, 640x480 dot addressable line displays with red, green, and blue dots are already in production. They can be battery operated, although large displays suffer from a limited contrast ratio and consequent limited viewing

angle and are flicker-free. Twisted nematic LC displays are being investigated as a solution to these problems and hold sufficient promise that they may be the battery-powered display of choice for future monochrome applications.

Other types of flat panel display are light-emitting diode, seeing renewed application due to a 5-fold improvement in luminous efficiency and to improved gallium arsenide processing, heterochromics, and electrophoretics displays. Electroluminescent panels use a transparent front panel, offering high resolution, light weight, and a narrow width. Flat panel displays have different characteristics from CRTs.

- a. Intensity Distribution. Plasma and TEFL technology use square pixels with a uniform density that results in a step-function intensity distribution, as opposed to the CRT Gaussian distribution. LCDs have an x,y matrix of electrical conductors that introduces a memory element that also works differently.
- b. Resolution. Flat panel displays have non-overlapping discrete elements and resolution is the number of pixels per unit distance on the display. This distance should be measured both vertically and horizontally with respect to the viewing distance, resulting in pixels per degree of visual angular subtense [Synder 88].

G.5.1.3 Projection Displays

Projection allows larger displays than either CRT or flat panel technology. 50-inch diagonal commercial systems are likely to become available within the next five years. Here again several technologies are being investigated, including LCs where, for example, LC transmissive mode is used for each color and then colors are combined for a single projection. Other approaches include light beam modulation (for example, using a laser display with rotating mirrors and galvanometers for x,y deflection onto a screen) and reflective displays based on mechanical mirror movement. In the latter case, prototypes with 2 million moving mirrors on a $1 \times 3/4$ -inch silicon wafer have been developed.

Both front and rear projection systems are used. A significant practical difference is that while front projection systems require a clear space between the projector and screen, rear projection systems can use folded optics to eliminate the blocking problem and, moreover, the projection of an image onto a transparent screen mitigates the effects of ambient light. On the other hand, rear projection systems require a larger volume. Rear projection CRT systems are favored for HDTV.

G.5.1.4 3D Displays

Typically 2D displays are used to provide 3D scenes in top, front, and side orthographic views. Hardware for 3D displays using either kinetic or stereopsis depth became available in the early 1970s. In the first case, a flat projection of a 3D scene appears strongly 3D when it is the projection of a rotating scene using, for example, arrays of lenses, holograms, or vibrating mirrors. Ideally, the depicted scene can be set in continuous smooth rotation. Stereopsis systems exploit the brain's ability to extract 3D relationships from the different views provided by two closely placed eyes. To this end, two slightly different views of a display are produced to resemble the different viewing perspectives of eyes set 6.5 centimeters apart. This may be achieved with two flat panels or time modulation on a CRT. It necessitates some way of separating the images presented to the eyes; in a workstation environment the user is typically required to wear special goggles containing shutters, or polaroid or red-green filters. In one commercial system, for example, passive circularly polarized glasses are used in combination with a stereo shutter, and a scene is displayed to one eye and rotated about a vertical axis by a few degrees before display to the second eye.

In virtual image displays, 3D images are projected in space. This is achieved through time and (virtual) space modulation, and a number of different approaches using both 2D and 3D displays are being investigated. For example, a rotating helix with a reflective surface is impacted by a laser beam, or an image is projected onto a vibrating speaker diaphragm with a reflective surface. The virtual display can be physically located in one place and optically superimposed on some conveniently placed surface and, thus, is potentially highly portable, helping to alleviate space and mobility restrictions in the operating environment. Consequently, the military is seen as becoming one of the major users on this technology. It is anticipated that future virtual image displays will exploit CRT and flat panel technology for image generation with holographic optical systems to provide high resolution, high contrast, and wide angle displays [Synder 1988].

Virtual user interfaces are expected to become available within the next five years. A current prototype system uses head-mounted displays which are implemented as two LCD TVs mounted a short distance (e.g., less than two inches) in front of the user's eyes. An optics system is used to expand the images so they cover most of the user's field of vision, and a diffusion panel to blur these images (so that the individual red, green, and blue dots in the displays are not apparent). The LCDs are driven by two NTSC video sources, perhaps using two graphics workstations synchronized over an ethernet with NTSC converters to

convert the video signals into NTSC format. An Isotrap digitizer determines the position and orientation of the user's head, providing information which is used to generate the images presented to the user.

While these technologies can provide good depth portrayal, they are expensive and often suffer from mechanical problems. Further, they incur very high-bandwidth requirements. It is estimated that the bandwidth necessary to create a true 3D image from a single source increases by approximately a power of 1.5 the requirements of an equivalent 2D image. Human factor concerns in such interaction still need to be investigated; in particular, how should 3D cursors be represented to enable natural 3D locating and pointing on a 2D display? How should 3D transformation parameters, such as rotation angle, be specified?

G.5.2 Input Devices

G.5.2.1 Keyboard

A keyboard is the primary alphanumeric input device. Keyboard size and packaging affects user satisfaction and usability. While the majority of current keyboards allow only one keystroke at a time (although dual key presses are used for capitals and special functions), chord keyboards that allow simultaneous keystrokes aid rapid data entry. Keyboards layouts are derived from early typewriter layouts. The QWERTY layout is still universally used, even though it has been shown to be far from optimal. Newer layouts include the Dvorak keyboard, designed to reduce finger travel distances, and the ABCDE layout intended for nontypists. Placement of additional keys, such as the shift key and number pads, also affects usability. More recently, keyboard designs that optimize wrist and hand placement have been developed.

Key design is another critical issue. Important design features include the force and displacement required for keypresses and tactile and audible feedback. Modern electronic keyboards use half-inch square keys separated by approximately a quarter-inch. Special keys, such as the enter key may be larger. When key position is important, physical locking in a lowered position or an embedded light can be used to denote position status. Special features such as a deep concavity or a small raised dot can be used to assure users of proper finger placement. Function keys are subject to several additional design factors intended to help a user remember the associated function. Cursor movement keys, another special cat-

egory, may have an auto-repeat feature with controllable speed. The importance of particular key types depends on both user profiles and the application. For example, function keys are often favored by expert users who readily recall the purpose of each function key, whereas cursor movement keys are particularly useful for form filling interfaces.

G.5.2.2 Pointing Devices

Several types of input devices are available for pointing. For the CCIS, pointing devices will be selected to reflect the communication styles, application, and type of data in question. This will promote device independence and portability since each logical classification can be implemented with a variety of hardware devices (although some are more convenient for certain kinds of data than others). The chief device classifications are *choice* for selecting menu options, *pick* for selecting components from a displayed set, *string* for specifying text and graphics input, *locator* for specifying a coordinate position, *stroke* for specifying a series of coordinate positions, and *valuator* for specifying scalar values. As an example of possible mappings between a logical classification and physical devices, the following discusses how various devices can be used to support each classification in graphics applications [Hearn 1986]:

- a. **Choice Devices.** A function keyboard that may be designed as a stand-alone unit or as part of a general use keyboard, where functions are programmable or fixed. A new type of device is a programmable display push button, with a dot matrix (typically 500 pixels) to display text and graphics. A touch panel or light pen is useful when menu selection is to be accompanied with spatial coordinate input; a joystick or mouse serves the same role. Keyboard entry requires menu options to be assigned a numeric value or some other form of short identifier. Voice input is best reserved for a small number of options.
- b. **Pick Devices.** A light pen is typically used to select an object or segment for some transformation or manipulation. A mouse, trackball, joystick, or tablet place control away from the screen surface and may be used along with a screen cursor to identify the position of the object. A keyboard is used by typing in the name of an object. Alternatively, buttons might be used, for example, by pressing one button to cause successive segments to be highlighted and another pressing another button to select a desired segment while it is highlighted.

- c. Locator Devices. When a thumbwheel, dial, trackball, joystick, mouse, or tablet stylus or hand cursor is used to position a screen cursor that specifies the desired coordinates, a button-push is required to input those coordinates. Light pens also serve as locator devices, for example, a small light pattern may be created for the light pen to detect. Keyboards are used as locator devices in two ways: coordinate values may be typed in, or special cursor keys may be used to move the screen cursor to the required position. Both methods are quite slow, and the first also requires the user to know the desired coordinate position.
- d. Valuator Devices. Valuator input depends on the application at hand and may comprise graphics parameters such as rotation angle and scale factors, or physical parameters such as temperature settings. A keyboard can always be used for directly typing in the required values. Special purpose devices, such as control dials or slide potentiometers convert linear movements into scalar values and may be more efficient.

Characteristics of interest include cost, durability, space requirements, weight, left or right-hand use, and compatibility with other systems. Accuracy is often a key selection factor; here a mouse or trackball is preferable when accurate pixel level pointing is needed. Pertinent human factors characteristics include speed of motion for short and long distances, accuracy of positioning, error rates, learning time, and user satisfaction. Some human factor aspects are still being researched; for example, the use of pointing devices that can be operated from the typing position, instead of moving to a separate device such as mouse. Here a force-sensitive device such as a isometric joystick positioned on the keyboard where force applied to the joystick is mapped to movement parameters such as velocity and position, can minimize time loss and user distraction.

G.5.2.3 Scanners and Digitizer Tablets

Special devices are used for inputting drawings and other graphics. Graphic digitizer tablets are precision measurement tools that electronically determine the physical coordinates of a point and transmit this coordinate data in digital form. They can function as both relative and absolute positioning devices, and also as menu tablets for inputting commands. Used to generate new graphics or copy existing drawing, these devices are increasing found in mapping, environmental, geologic, medical, and industrial fields as well as their traditional CAD applications. A tablet may range from a few square inches to drafting-board

size and is accompanied by a transducer, either a stylus pen for free hand drawing and cursor control on the monitor, or puck cursor for more accurate positioning and greater flexibility. Electromagnetic, sonic, electrostatic, or magnetostrictive technologies are used to establish a relationship between a tablet surface and the transducer and four operating modes may be provided (point, switch stream, continuous stream, and incremental). Resolution specifies the smallest distance that can be detected in the digitizer's output, normally ranging from 100 to 1000 lines per inch, or 0.01 to 0.001-inch. Additional key characteristics include the specific data formats handled by the digitizer and options provided for setting operating parameters such as data rate, origin location, and incremental limit.

Scanners are the other major tool for entering a drawing. These are optomechanical devices and convert a hardcopy image into a raster drawing that can be used by a variety of graphics software. After scanning, raster-to-vector conversion may translate the bitmapped image into vector form for further manipulation. In addition to CAD/CAM, mapping, and geographical applications of digitizers, scanners support data storage/retrieval and technical publishing. Key requirements are resolution (ranging from 100 dots per inch (dpi) for low resolution approximation through 200 to 400 dpi for accurate reading of line drawings up to 1000 dpi for fine line maps) and speed (from 1 to 24 minutes for an E-size drawing).

G.5.2.4 Touch Technologies

Touch technologies have the potential to support multiple virtual devices simultaneously, largely because they do not need handheld intermediate transducers. The absence of moving parts offers durability in high-use environments. Touch screens provide a direct connection to the display through some overlay mechanism using such technologies as conductive and capacitive films, infrared sensing matrices, or plastic overlays with embedded crossing wires. Although widely used, for example, in public information systems, early devices exhibited poor precision, slow and erratic activation, and poorly designed displays. Several new techniques, such as a *two-touch* strategy, are being used to overcome these problems. *Lift-off* strategies are another innovation and allow higher precision by showing users a cursor on the screen slightly above their fingers. The cursor can be dragged continuously across the screen and functions are activated when users lift their fingers off the surface, sometimes called the *untouch screen*. Dragging icons and other objects allow touch screens to support direct manipulation interfaces. Sensing is with the finger, allowing physical templates to be placed over the device and provide kinesthetic feedback, although a

physical device on the display surface may be damaged and become dirty resulting in image degradation. Infrared devices that are implemented by a frame around the display surface do not suffer the disadvantages of a physical overlay, but may have optical parallax.

Direct manipulation interfaces often employ virtual devices such as buttons and switches. Although any input devices that are position sensitive and employ a fixed planar coordinate system can be used, touch tablets permit a large surface to provide textual guidance to novice users and partitioned touch tablets are currently seen as the most promising option for virtual input devices [Brown 1990]. Also, they can employ acoustic, electronic, or position sensing techniques to allow operation by placement of a finger, pencil, puck, or stylus. Prototype touch tablets with the ability to sense multiple points are under development. Moving such controls from the screen to corresponding virtual input devices at the work surface frees up valuable screen space.

G.5.2.5 3D Input Devices

Currently the interaction and manipulation of 3D graphics is primarily 2D in nature. The input device is typically a mouse, dial, or keyboard providing, respectively, a flat, linear, or discrete input medium. 2D input from a tablet or triad mouse can also be converted to 3D space. For example, tablets can map x,y hand translations to object rotations about y and x, and map a stirring movement to rotation about z.

Placement in space is a six-dimension (6D) operation requiring three variables to specify location and three more to specify orientation (yaw, pitch, and roll). Accordingly, new input devices are coming into use. These include multi- and three-axis joysticks, three-axis track balls, and number wheels. One example is provided by a variant on a conventional mouse, called a Bat, that enables the usual relative positioning but delivers data 6D [Ware 1988]. Other high dimension devices include data gloves which can provide 3D gestures or, more recently, data suits. Here feedback can be provided by echoing user actions in a virtual environment as animated input, input to affect virtual controls, or direct handling of virtual tools; force feedback is under development. These multiple dimension input devices are relatively inexpensive and continued improvement in resolution and bandwidth is expected. One of the basic underlying technologies is a spatial monitor. The McDonnell Douglas Polhemus 3Space Isotrak, for example, is a six degrees of freedom spatial monitor that signals orientation and position relative to a fixed source. A sensor yields 9-bits of res-

solution in each of the six variables and a digitizer can be used to determine the position and orientation of the input device through special fiber optics. There are some problems associated with the use of an Isotrap digitizer; for example, it broadcasts an electro-magnetic field that requires multiplexing to handle multiple devices. It also has a limited range of accuracy which deteriorates in the presence of metal objects and other sources of electro-magnetic radiation, such as fluorescent lights and computers [Green 1989; Kruger 1983].

In conjunction with a 3D display, these input devices yield a 3D user interface. One example of a practical 3D user interface is a 3D versatile volume visualization system. A 3D positioning and orientation Polhemus input device, called a *kite*, is used to handle input events with feedback in the form of a 3d cursor called a *jack*. A mouse and keyboard are used for traditional command and pick selection and a conventional 2d work surface is used to present a small-scale virtual 3d reality. Instead of pixels, the system uses *voxels* for volume elements (and a *voxblt* operation instead of rasterops, see section G.6.1), *figurines* for icons, and *rooms* as the 3d equivalent of windows.

G.5.3 Video Devices

The primary video input devices are the digitizing devices previously discussed, cameras for both motion and still video), and videotape recorders. Videotape recorders are also used for output along with video printers. Time has not permitted an in-depth review of available devices but a discussion of the critical issues in interfacing video devices with a graphics workstation can be found in Knierim 1989, 440-442].

G.5.4 Hardcopy Devices

G.5.4.1 Printers

Various printers are available for producing paper documents that can be copied, mailed, marked, and stored. Important criteria in the selection of such devices are speed, print quality including features such as highlight techniques are available, cost, such as that incurred by requirements for special paper, environmental considerations such as compactness and quiet operation, support for special forms, and reliability.

Different types of printers exhibit different combinations of characteristics. For example, dot matrix printers can produce upward of 200 characters per second using multiple fonts with boldface, variable width and size characters, and graphics capabilities. Daisy wheel printers produce high-quality characters but at lower speeds, typically 30 to 65 characters per second. Inkjet printers also offer high quality output, this time in conjunction with quiet operation. Thermal printers are quiet, compact, and offer relatively inexpensive output on specially coated paper. Laser printers operate at 30,000 lines/min and are widely available for microcomputers with many software products providing publication-quality typesetting for desktop publication. Desktop laser printers offer a resolution of 300 dpi. Graphic images are typically large in size, requiring additional computing and memory capacity and longer imaging times. Because of the inherent complexity in outputting a font, fonts and type face are an important issue. Most laser printers offer a variety of standard fonts whose geometry is embedded within the electronics of the printer to allow fast loading and output of those fonts. The next class of printers offer 600 to 110 dpi. These output between 360,000 and 1.2 million dots per square inch, increasing both required computing power and memory. Ultra-high resolution printers offer 2500 dpi which is the equal of typesetting. Most printers employ a raster image processor (RIP) that takes workstation output and converts it to a language the printer can understand. The RIP may be built into the printer or provided as a separate processor. The problem of compatibility that this raises is being addressed by the industry and PostScript has emerged as the industry standard.

G.5.4.2 Plotters

Pen plotters are widely used for precision drawings with smooth edge lines. The primary types are drum plotter, often using a friction drive to handle media in both cut sheet and roll form, and flatbed plotters that hold the plotting media stationary on a flat surface while a pen moves horizontally and vertically. Pertinent characteristics include:

- a. **Media Size.** The size of paper media handled. This can be specified in terms of the ANSI measurements A-size (8.5 x 11 inches) to E-size (36 x 48 inches) or the equivalent ISO metric sizes A4 to AO.
- b. **Throughput.** A measure of the time needed to complete a drawing, affected by pen speed, pen up/down time, pen exchange time, and acceleration as well plotter-computing speed.

- c. Resolution. Mechanical resolution is the smallest move a plotter can make in any direction, generally ranging from 0.0005 to 0.005-inch. Addressable resolution is the smallest move a user can address.
- d. Accuracy. The ability to move the pen an exact distance or move to a specified point, and is affected by paper, pens, or environmental factors such as humidity.
- e. Repeatability. The ability to replace the pen precisely on a point previously drawn.
- f. Velocity. The speed on pen movement. Typically measured in inches per second (ips) or millimeters per second (mm/s), velocity may vary with line direction and ranges from 1 to 32 ips.
- g. Acceleration. The time it takes for the plotter pen to move from initial to maximum velocity, given in meters per second² or gravities. Acceleration may be constant over the entire acceleration phase or stepped, and is particularly important for small line segments.
- h. Pen Number and Types. The number of pen holders provided (these may allow variable pen configuration), and specification of whether pens employ liquid ink, ballpoint, fiber tip, or ceramic, and specification of color and line width.
- i. Command Sets. The commands provided for generating, for example, characters and arcs. As yet, there is no well-accepted standard for plotter commands. Data sorting eliminates the need for embedded plot optimizers.

Alternative technologies employed are electrostatic, ink jet, and thermal transfer. Electrostatic hardcopy devices generally use a fixed array of electrodes to deposit electrostatic charges directly onto a dielectric surface, and the image is then toned, developed, and fixed. High voltage and switching frequencies are required. While electrostatic devices can theoretically provide a throughput of 1800 pages a minute, in practice the use of multiplexing schemes and mechanics reduce this to one ANSI E-size page per minute. They provide color capability, the ability to handle large format reproduction, and a printing feature but the need for dielectric-coated paper incurs additional support costs.

Ink jet devices can also produce large-format color output. The two major design approaches are continuous flow and drop-on-demand. While ink is pumped through the nozzle only when a mark is needed for drop-on-demand, in continuous stream approaches ink is continually forced under high pressure through a controlled small nozzle, and the ink

stream must be deflected from the paper when no image is desired (either into a gutter or by using highly charged droplets that repel each other). High-speed ink jet systems may use an array of individual jets extending over moving paper, moving the jet assembly over stationary paper, or maneuvering the ink stream across the imaging area. Again theoretical speeds, in this case image generated at a square inch per second, are lower in practice. Resolution ranges vary between 80 and 300 dpi.

Thermal transfer systems pass plain paper accompanied by a sheet coated with pigment-impregnated wax under an array of individually controlled heating elements, each corresponding to a single pixel. As the wax is heated by the elements it migrates onto the paper although the difficulty in controlling the amount of wax transferred presents problems for shading. Color is generated by using a separate transfer sheet for every primary color, requiring successive overwritings to produce additional colors. Throughput is limited by cycle time for the heating elements, and for a printer with a page-wide array of heating elements it is in the range of five single-color pages per minute. New fabrication techniques are increasing resolution from around 200 to 400 dpi. The major drawbacks include size format restrictions and the high cost of supplies.

Photographic printers are used for creation of 35 millimeters, or larger, slides and photographic prints. These may be an add-on device in front of a display or an independent high-quality printing system. Computer output to microfilm devices is effective for high volume output applications.

G.6 GRAPHICS HARDWARE

Traditional workstation hardware consists of a processor, memory management hardware, memory, and I/O devices including a mouse, keyboard and monochrome display. Special graphics hardware can be added to increase graphics processing power.

Industry groups and individual companies are attempting to develop standard methods for evaluating workstations in general, and the performance of graphics products in particular. The Graphics Performance Characterization (GPC) group, a commercial consortium, is pursuing the development of a four-level standard:

- a. Primitive Level. Describes raw capability in units such as polygons per second.
- b. Picture Level. Describes the rate of graphics output.

- c. System Level. Deals with I/O and image manipulation.
- d. Application Level, a test suite of typical functions on public domain images.

G.6.1 Bitmaps and Rasterops

Bitmaps allow both off-screen generation and storage of images. The use of additional bits for display of color or intensity variations can result in high-storage requirements for the bitmap. For example, a system employing 24 bits per pixel, with a screen resolution of 1024 x 1024, requires a bitmap with 3 MB of storage. Depending on the application, there are several approaches for encoding intensity information to reduce this memory requirement. While chip manufacturers are announcing products which improve the performance and reduce the cost of bitmapped displays, bitmaps are device dependent.

The fastest way to refresh a screen image is to maintain the complete image off-screen in a bitmap and move it back onto the screen using a single block operation. RasterOp, or *bit boundary block transfer (BitBlt)*, is an operation for moving a rectangular bitmap from one area of memory to another. The display may have pixels visible in the processor's address space, with RasterOps performed by the processor (possibly with special hardware assistance). Alternatively, the pixels may not be addressable by the processor but manipulated by a separate RasterOps processor. The performance of the RasterOps processor is bound by the display memory width (even if done entirely in software), although microprocessors with barrel shifters can be used to improve the performance. Higher-level window RasterOps can be used by a window management system for moving information, image creation, window movement, and changing window attributes such as size and covering.

G.6.2 Colormaps

One of the special processing concerns associated with the use of color is the need for *color look-up tables* or *colormaps*. There are two kinds of color storage. In the first case, each entry in a bitmap stores a value that is an index into a colormap determining the color (or grey shading) displayed at that location. The translation of an index value into an actual color is performed by the display driver hardware for the screen and is transparent to the user and programmer. Since common raster CRT devices have red, green, and blue guns for producing color, colors in the table are usually specified by their green, red, and blue (RGB)

values. Just as pixels are a real source to be shared among computing clients, so are the entries in the colormap. Applications should be able to use a number of different pixel values and corresponding colormap entries without being aware of other windows also using the colormap. Hardware assistance for this sharing is useful. In the second type of color storage, *true color*, each bitmap entry holds the actual RGB representation for its color. This is more expensive, requiring three or more bytes of memory for each pixel.

Color graphics software may allow a user to alter colors by displaying a color palette on the screen. For example, Tektronix's TekColor System contains the HVC model to aid in the selection of colors from among 16 million possibilities and a device-independent color editor for X, called Xtici, that provides an interactive color interface to the HVC color space for quick specification of a color palette [Toole 1989].

Video display devices incorporate VGA-compatible colormaps with a color video controller, and a range of transputers for video-computation and data manipulation tasks in multimedia applications.

G.6.3 Graphics Accelerators

Graphics accelerators originated to overcome inadequate computer speeds for rendering or manipulating complex images in real time by applying dedicated hardware to graphics display problems. They are designed with parallelism in mind, support a large physical cache, and generally use a shared memory multiprocessor to provide a floating point performance on the order of ten times faster than that of basic microprocessors, enabling an overall performance approaching that of current generation workstations with a graphics accelerator. Geometry and imaging accelerators aid different aspects of graphics and have greatly differing memory requirements and architectures. Geometry accelerators typically generate pixels that are written into a display memory of 1280 x 1024 resolution. Their performance is measured in terms of the number of geometrical objects that can be manipulated in real time. For example, one popular geometry accelerator has the ability to generate about 5,000 shaded geometric polygons per picture in real time (defined to be 10 to 30 pictures per second). Imaging accelerators are specialized to compute on images only and are measured in terms of the number of pixels they can comfortably manipulate on the order of 100 times faster than a host computer, potentially attaining a computational power of 200 to 600 times that of a VAX 11/780 on this class of computations. They include large image

memories that are directly addressable by the accelerator. A new type of accelerator expected to appear within the next few years is a *rendering accelerator* which accelerates the non real time aspects of geometry which the geometry machines are incapable of handling. Visualization accelerators are mentioned in Section G.7.1.

G.6.4 PC Graphics Boards

Graphics boards are evolving into three different families of products. *2D Drawing Assist* boards support a host processor in such operations as 2D graphics, vector drawing, clipping, bit manipulation, and vector scaling. The *Shading Assist* boards provide assistance in shading algorithms. Finally, the *3D Assist* boards perform 3D transformations, surface calculations, and ray tracing calculations. The next generation of boards will employ multiple processors which connect like a transputer or by combining mixed processors (e.g., RISC and digital-signal processing devices).

Such boards provide better resolution, colors selection, and performance to extend the capabilities of graphics processors. The new communications bus interface standard Extended Industry Standard Architecture (EISA) offers improved communication by replacing the Industry Standard Architecture (ISA) bus interface used in most PCs with improved direct memory access, 32-bit transfer, and the ability for a graphics board to share host memory for display list storage at high speeds. Improvements in the amount of memory per package and the physical space used by the memory are allowing more memory per board. This increased memory can be used for improved resolution or doubling of buffer video memory to support animation. The increasing use of frame buffer standards (for example, medium resolution VGA) will allow these boards to act as graphics data managers for the host. Very high performance microprocessors with graphics cards are evolving to the extent that PC graphics capabilities are beginning to approach the look and performance of workstations. Already user interfaces with automatic sensing and adjustment to the available resolution of a display have been proposed. As costs continue to fall, these boards will move from dedicated CAD applications to broader PC markets.

G.6.5 Graphics Workstations

Scientific computations, such as computational fluid dynamics and mechanical analysis, are floating point intensive, dealing with tens of megabytes of data. Accordingly, graph-

ics workstations are in transition from simple integer machines with graphics hardware to fast floating point machines that can perform analysis and simulation as well as display the results graphically. Smooth animation, for example, requires the interactive display of colored, shaded, 3D geometric graphics, as well as pixel based image data. Generally, at least 10 frames per second with data transfer rates from main memory to the frame buffer of at least 10 MB (MB=millions of bytes) per second are required to animate a sequence of 8-bit images [Borden 1989, 279], although frame rates on the order of 3 to 4 frames per second may be considered acceptable for medium to low range systems. Minimum system requirements are typically on the order of 20+ MIPS, 20+ sustained MFLOPS, 64 MB of real memory, and 50,000+ shaded polygons per second. Main memory requirements for a workstation are on the order of 25 to 100 MB for data storage, exclusive of the operating system and application software requirements, with a total frame buffer memory requirement of 3 to 8 MB. The frames (typically hundreds of megabytes of data) may be stored on either a local disc or remote discs on other networked machines. Local workstation disk storage on the order of 250 MB to 1 gigabyte provides a reasonable balance between rapid data availability and the economics of central data storage.

New VLSI processors, providing high performance floating point computation rates and 3D shading hardware, have been introduced to enable cost-effective rendering involving trace images and the use of multiple light sources to allow the entire 3D graphics pipeline to be implemented in software. Such processors enable a new class of workstations with computer rates previously associated with supercomputers and visualization super-workstations at the price of 2D workstations.

A new class of computer, called a graphics supercomputer or super graphics workstation, combines a RISC architecture processor, tightly coupled 3D- and pixel-based graphics, bus technologies, and custom integrated circuits. The goal is to display moving images, or other time-dependent information about relatively complex systems, in real time. The several types of machines commercially available all embody different architectures. Some use vector floating point, incurring additional design requirements. An 8 MFLOP sustained computation rate requires 64 to 196 MB of memory bandwidth. Memory must be interleaved and accessed efficiently with a stride greater than 1 to operate on data which is not packed sequentially in memory. Since tens of MB of real memory are active in these systems, the odds of a single bit error are so high that memory must be protected and corrected in order to sustain reasonable reliability. Vector processing does not apply to all applica-

tions, however, and parallel scalar processing may be preferable in some circumstances. Future high-performance workstations may have both vector and scalar processors.

By 1995 sustained floating point performance will be on the order of 600+ MFLOPS. Systems will have 300+ MIPS per processor and will support up to 4 or 8 integer, scalar, and vector processors. Main memory bandwidth will be 128+ MB per second with at least 16-way interleaving. Graphics performance will be over one million polygons per second transformed, clip tested, light-source shaded and drawn. Hardware will support transparency, depth-cuing, textures, and other special effects.

G.7 SPECIAL APPLICATIONS

G.7.1 Visualization

The term visualization was first popularized in the National Science Foundation's (NSF) landmark report *Visualization in Scientific Computing* (ViSC) [McCormick 1987]. It refers to an extension of the concept of multiple windows to multiple views, where each view provides a different graphical representation of the same underlying objects and processes and users can communicate with data by manipulating a visual representation of that data. Several visualization tools are available commercially or in the public domain (for examples, see [Dyer 1990]). Visualization arose to meet the need of understanding and interpreting the vast amounts of data produced by supercomputers, network computational servers, and data generating devices such as satellites, 3D sensors, and other high bandwidth devices. For example, synthetic aperture radar may have data recording rates on the order of 10 to 100 megabits per second and National Aeronautics and Space Administration's (NASA) Earth Observing System, planned for deployment in the late 1990's, will have to receive, process, and store over 10^{12} bytes per day. Data may be collected for analysis in near real time or at a later time.

The ability to use real images provides an effective communication tool as well as a decision influencer. One example of a possible type of CCIS application is interactive visual analysis of technical data such as occurs with remote sensing used for satellite imagery and geographic data representation, for example, analyzing current reconnaissance imagery to evaluate battlefield information such as troop strength, readiness, level of supply, and intelligence information. ViSC can be used to support desktop video production, employing a

variety of display techniques to convey large amounts of information including 3D graphics, image processing, photorealistic rendering, and volumetric display together with techniques such as stereo displays to enhance 3D data representations. Animation is a major ingredient, providing sequences of images that illustrate the interrelationships of various parameters over time, for example, to show simulated flights through a terrain.

Visualization places special requirements on user interface handling and processing. In the first case, a common user interface for the selection of the techniques for representing data and associated options, together with a common design for displays, is required to allow software to manage arbitrary data sets. Common data access standards are required to enable application independence, for example, to support correlative data analysis, where data from a variety of sources is used to study a problem. Discipline independence may also be required. The management of, and access to, data must be decoupled from the actual visualization software to provide a clean interface between the data and its display based only on some basic descriptions of the data. Significant processing requirements arise from dynamically changing graphics, for example, interactive video systems used in training and simulation require stored imagery to be merged and displayed with graphics generated in response to user actions, and both real imagery and generated graphics can change at rapid rates. Technical data analysis often requires combining several 3D data sets to portray four or more dimension, for example, using light shading on rendered surfaces to allow a series of surface images of a rendered data set to be created with slightly differing viewing angles. Such flexible combinations of functions incur heavy computational costs. Fast data paths to data generation and storage devices, fast computation dedicated to preparing data for display and computing pixels, and a flexible display system to support the various display formats are required. Although modern high performance workstations have aided the development of visualization, even augmented with traditional accelerators, they cannot meet these demands. Instead, a *visualization accelerator* is often required. The functional and performance requirements of this type of accelerator are outlined in [Whitton 1989, 331-339], along with a recommended architecture. Although commercial visualization accelerators are available, there are as yet no standardized figures for the processing speeds of the hardware components or the visualization system as an integrated unit.

The application of networking to ViSC, called *televisualization*, requires major enhancements over existing networking capabilities including, for example, increased data rates and compression and decompression algorithms to speed up the transmission of visu-

alization data. The emerging High Speed Channel Standard Interface from the ANSI X3T9.3 Committee will be essential for the integration of high performance distributed systems needed for real time interactive visualization. National gigabaud networks proposed for the late '90s will allow remote user access to interactive visualization.

There are many outstanding issues. For example, in volume visualization there is a lack of algorithms for rendering lines, curves, surfaces, and volumes into volume memories and general utilities for arbitrary rotation, changing volume size, and hidden volume removal are still needed. Other issues include determining how much graphics processing should be done on the supercomputer and how much moved to the workstation. Several of the large scientific research centers are conducting ViSC research [Rosenblum 1989, 68-83]. For example, the National Center for Supercomputing Applications' Rivers Project is conducting research to develop hardware and software systems for interactive 3D visualization and interactive steering of supercomputer simulations in a high performance distributed environment. Its Software Development Group develops software packages and X support to provide visualization capabilities to remote users, the recently established Renaissance Experimental Laboratory provides a classroom for supercomputer and visualization education, and the Visualization and Media Services Group produces film and video output for graphics generated during scientific research. The Naval Research Laboratory (NRL) is examining the ability of visualization to portray complex simulation data in real time in support of its electronic warfare studies and in physical oceanography to support global climate modeling in naval systems for surface operations and antisubmarine warfare. Industry and academia are also performing visualization research. In May 1990, academia and industry joined in establishing the Institute for Visualization and Perception at the University of Lowell. A group within the university is chartered to advance technology in the fields of data visualization, graphics, and auditory data representation and user interfaces.

G.7.2 Hypertext

The terms *hypertext* and *hypermedia* refer to a network of information nodes that consists of textual or other objects, *entities*, that must be manipulated, *links* that encode static and dynamic information, and *properties* of those nodes and the links between them. Their primary role is to provide better utilization of computers by supporting the ability to browse information freely. Links may be typed to provide extended functional capabilities and, with node typing, this permits the definition of a grammar that facilitates user navigation

and automatic information retrieval. Labelling can be used to provide representation of, for example, constraints, timing, inferencing, or rhetorical information. The disciplined use of structured composites of nodes and links as higher-order entities is a fundamental principle. A hypertext system integrates graphics technology, primarily windowing, and text to allow users to manage and access information. The essential concepts that distinguish these systems from more traditional document support systems are data type and media independence, absence of a hierarchical structure, and distinct separation of form and content. Linkage between objects arbitrarily located within a large multi-topic and extended-history document is typically provided with support for creating, studying, organizing, and linking within and between the many overlapping and nested knowledge domains.

Current uses focus on in-house and external technical documentation and distribution of multimedia materials, particularly on optical disks. Another potential use is as a mechanism for collaborative work in the computing environment. In this latter case potential applications range from dealing with policies, procedures, and regulations to collaborative problem-solving. User interface issues are concerned with constructing links among nodes and with browsing a completed network, particularly with respect to the form and manipulation of the display and navigational mechanisms. Many issues remain to be resolved. For example, whether information should be divided into screen-sized chunks or whether the screen should be treated as a window that moves over a larger unit of information. Since there appear to be applications where each approach is superior, it may be best to accommodate both.

A layered architecture with published interfaces will aid the access of a hypertext system by others and, hence, the development of systems that can interact with each other, as well as users, across a variety of local and wide area networks. Possible layers are:

- a. Data Layer. Provides consistent data management for all information in the hyperbase, including both node contents and links.
- b. Element Layer. Provides separate services for managing nodes and links, and translating the raw data of the data layer into these elements of hypermedia.
- c. Inference Layer. Provides the ability to traverse a link and review the nodes at the destination.
- d. Interface Layer. Defines mechanisms through which the user interacts with the hyperbase and is responsible for displaying information contained in the nodes.

The basic components for a hypertext system comprise a *user-interface toolkit* where the presentation layer should be independent to allow structure and content to be displayed in many ways; a *domain-specific data model* specifying a uniform structure for nodes, relationships, and content, thus defining what the hypermedia system can represent; a *type and object manager* where media types provide primitive representations for text, bitmaps, video and audio, and graphics; and persistent *storage modules* where the mapping of information to permanent storage should be independent of what is represented. Well-defined interfaces between these modules will aid independence. For example, queries and indexing should only relate to whether there are sets, collections, or other navigation paths to iterate through and whether there is cached information (indexes) that can be used to limit the search. Whether a system is multiuser or distributed should be transparent to a user.

Standards applicable to different media types are evolving separately and hypertext standardization efforts must interact with other media and document models. Hypertext reference and data models that will provide a basis for new standards are under development [NIST 1990]. Whereas data exchange on the document level can be approached through adherence to various file format standards, interchange standards for transferring data between similar systems are also critical. One proposed standard is the X3V1.8M HyTime DTD interchange format.

Future standards should support distributed hypertext within an open systems framework. They should ensure interoperability in current computing environments and transition to succeeding generations of technology. In particular, policy neutrality and extensibility to support rapid evolution of both data type specific software and notions of usage of links are important for evolvability.

G.7.3 Teleconferencing

Communications can occur in many forms using a wide variety of methods. For example, in a networked environment, one of the first applications typically deployed is electronic mail. Electronic mail allows users to communicate. The message creation component consists of a text editor. The user composes the complete message, which is subsequently given to the mail system for routing and delivery. The mail system electronically transfers the message. The recipient views the message completing the communications loop.

Electronic mail captures, transfers, or communicates formal messages. It does not capture the morphology which can be important in some forms of communications. Extending electronic mail, to include audio, would capture the message and its associated morphology. However, augmenting mail to include video captures non-verbal, as well as, verbal communications. Teleconferences extend this concept, audio and video communications, to interactive conversations or real time. Although teleconferences are an effective communications technique, and the technology has been available for some time, it has not achieved widespread use. The limiting factors have been cost related. These factors are being addressed and teleconferences will become a viable communications technology by the 1995-1997 time frame.

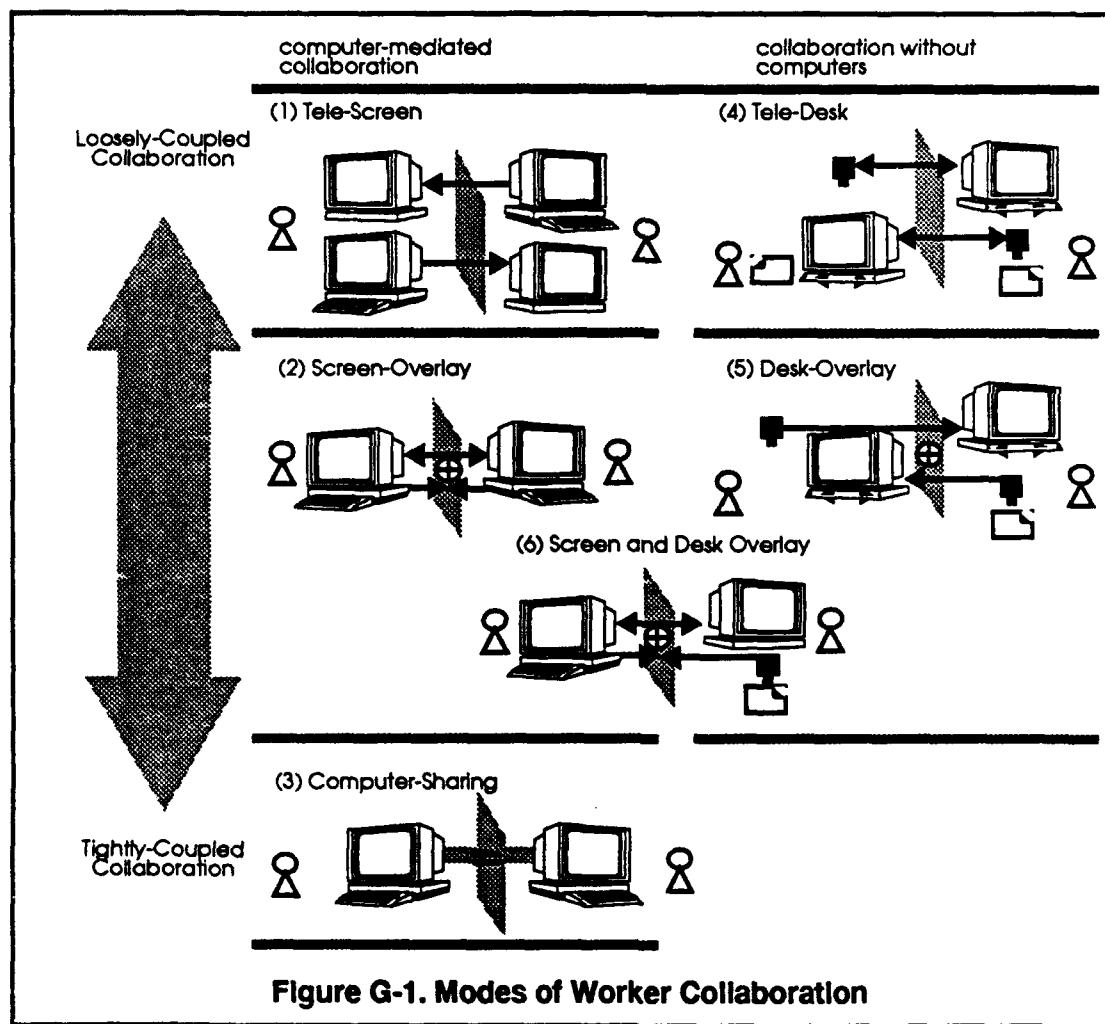
Teleconferences requires hardware to capture the video and audio portions of the message. Video cameras and video cassette recorders have recently become consumer products. This hardware provides the functionality needed for teleconferences. Hence, an inexpensive COTS item can be used to capture the video and audio portions of the message. The presentation of the message has not been a problem. A standard television could be used to present the message. However, the technology of computer workstations allows other solutions. It is not uncommon for a base model computer workstation to include the capability to present audio messages and although it may not directly support video, optional hardware will allow video to be presented within a window. Hence, presentation portion of a teleconference is possible via COTS items.

A factor that has limited the implementation of teleconferencing has been the ubiquitous access to a cost effective network. Integrated Services Digital Network (ISDN) and data compression technology will provide the network capabilities required for teleconferencing. ISDN provides the network access needed for teleconferencing applications. Data compression reduces the bandwidth requirements of teleconferencing applications, for example, the bandwidth requirements of a low bit rate, quarter resolution teleconference can be reduced to 64 Kbps.

G.7.4 Desktop Conferencing

A concept, one that permits general-purpose group interfaces, is that of video and audio *virtual shared workspaces*. Individual workspaces are overlaid in a virtual shared workspace, together with a shared drawing surface. Users are able work in either individual or

collaborative modes and the ability to move easily between these is important. In individual mode, users should be able to work independently of other members' actions. Current desktop conferencing systems allow participants to hold real time conferences by interchanging information through video, voice, and multimedia documents, using a common window management system such as X, electronic writing pads, loudspeakers with microphones, image scanners, video cameras and video processors. For example, the Team Workstation prototype under development by NTT Human Interface Laboratories supports three levels of collaboration and six collaboration modes as shown in Figure G-1 reproduced from Ishii [1990, 19]. The tele-screen and tele-desk modes are for the non-overlaid remote display of



individual screens and desktop respectively. They are useful for showing information to remote users in a loosely-coupled environment. The computer-sharing mode is for tightly

coupled collaborations such as coauthoring. The tele-desk, desk-overlay, and screen and desk-overlay modes allow sharing computer-based material and pointing/drawing gestures on workers' desktops. Team WorkStation also is an example of the types of network communications needed to connect distributed workstations. The current prototype uses a combination of RGB and video, voice, input device and data networks in a Macintosh LAN. Future versions will employ multimedia LAN and ISDN.

Architectural questions are being examined through the development of another prototype, the Multimedia Environment for Remote Multiple Attendee Interactive Decision-making (MERMAID) system [Watabe 1990, 27-38]. This system uses a server-client model to provide the flexibility required to support a variety of applications. Clients supply participants with user interfaces for smooth interaction with the system. Servers supply functions for accomplishing group collaborative work such as managing conference progress, handling conference information, local area network support, and controlling transmission routing and information flow among collaborators.

A key user interface question is how to design aids and interfaces that will enable users to track and handle their increasingly complex multimedia environments. Display issues are also relevant, for example, should individual and shared images be overlaid or tiled on a single display or produced on adjacent displays? If video overlaying is done at a standard video signal level, heterogeneous computers can be used. In addition, there are dynamic and complex interrelationships between interactive media and information intensive work where new qualitative phenomena arise, for example, understanding cooperative intelligent behavior when cognition is distributed over dispersed group members and intelligence is shared among persons and machines.

Technical issues that must be addressed include, for example, how to manage distributed databases with which multiple user groups at different organizational levels and locations may interact and how to develop and support applications that are modified and shared by working teams. Networking is also important. Some organizations are preparing to support this type of cooperating working with a multimedia infrastructure, that is, by completely wiring a building with audio, video, and data networks. In such a case, audio-switching between rooms can be under computer-controlled from any workstation, giving each member the ability to make arbitrary audio-video connections throughout the facility

G.7.5 Audio/Video Telecommunication

Information interaction that allows people to engage in informal, unplanned, and unstructured interactions cannot completely replace the social and organizational integration of team members, but can fill many of the communication gaps that arise in large or dispersed teams. Such informal interaction seeks to allow direct interaction among people with the technology relatively invisible. The emphasis is on audio/video communications to emulate proximity. Unlike computer conferencing, the goal is to support interactive meetings with unscheduled random participation, unplanned agenda, and a high bandwidth content where no special actions are required to establish a conversation. For example, a visual channel may be used to support identifying a partner for conversation, identifying when a potential partner is available, and establishing a topic for conversation in addition to serving as a communication channel. This allows a simple human interface where someone at one end of the link simply has to speak and they will be heard at the other end. Of course, the positioning of the system is crucial to its success.

Essential architectural requirements include a very high aspect ratio video channel and full duplex teleconferencing; technology not fully within our grasp. A wide image to convey a heightened sense of realism compared to a traditional television image is needed and, consequently, aspect ratio is an important characteristic. One example groupware system, VideoWindow, employs a screen size 3 feet high by 8 feet wide to produce images that are approximately life-size and allow a broad area to be visible. The audio system provides four independent channels arranged to maintain the spatial localization of speakers and other sounds. Equalizers, highly directional microphones, and some signal processing techniques are used to prevent ringing and auditory feedback. Audio switching arrangements in standard teleconferencing setups are used so that only one person at a time can speak. Transparency is another important issue and camera and microphone placement are critical. Even so, current technology-mediated systems do not have the communications flexibility and ability to manipulate media characteristics that are automatically handled in face-to-face communications, for example, conversation regulation by eye contact suffers.

G.8 REQUIREMENTS FOR HUMAN FACTORS ENGINEERING

The size and flexibility of the system envisioned for the year 2000 and beyond makes it important for the user interface to be optimized with respect to considerations of human-

factors. Indeed, the test and evaluation plan specifies that the "WAM program will adequately address Manpower and Personnel Integration (MANPRINT) areas of concern to include human factors engineering, safety, health, manpower, personnel, and training" [WAM TEMP 1990]. At its best, proper attention to human systems integration in the conceptualization and design of a CCIS will provide benefits in five domains:

- a. Gains in efficiency and effectiveness through improved human performance (e.g., accuracy, ease, safety, and speed).
- b. Lessening of the personnel requirements for system operations and maintenance (e.g., the numbers of needed people, special aptitudes, and skills).
- c. Reduction in training costs (e.g., the equipment, manpower, time, and dollar costs).
- d. Avoidance of equipment and personnel losses (e.g., from accidents, exposures to health hazards, or human errors).
- e. Improvements in the comfort and acceptance of all users operators and maintainers.

This section is focused on the *human factors engineering* aspects of the larger issues of human systems integration in the target architecture for a future CCIS.

G.8.1 Applicable Documents

The architecture, design, and implementation of any future CCIS should conform to the latest version of certain relevant Government-issued specifications, standards, and handbooks, as listed in the Department of Defense Index of Specifications and Standards (DODISS), and supplement(s).

Specifications:

- a. MIL-H-4685SB Human Engineering Requirements for Military Systems, Equipment, and Facilities.

Standards:

- a. MIL-STD-12 Abbreviations for Use on Drawings, Specifications, Standards, and in Technical-type Publications.
- b. MIL-STD-411 Aircrew Station Signals.

- c. MIL-STD-490 Specification Practices.
- d. MIL-STD-783 Legends for Use in Aircrew Stations and Airborne Equipment.
- e. MIL-STD-1472 Human Engineering Design Criteria for Military Systems, Equipment, and Facilities.
- f. MIL-STD-2167 Defense System Software Development.

Handbooks:

- a. MIL-HDBK-759 Human Factors Engineering Design for Army Material.
- b. MIL-HDBK-761A Human Engineering Guidelines for Management Information Systems.
- c. MIL-HDBK-763 Human Engineering Procedures Guide.

National and international human-factors guidelines and standards have been published or are under continuing development. A future CCIS should also conform to published guidelines, standards, and handbooks to the extent that they may represent the latest state-of-the-art in good design and use practice, whether or not listed in the DODISS. Examples include:

Guidelines:

- a. Boff, K.R., and J.E. Lincoln (eds.) "Engineering Data Compendium: Human Perception and Performance." Volume 1. Wright-Patterson Air Force Base, 1988.
- b. EPRI NP-3659 Human Factors Guide for Nuclear Power Plant Control Room Development. Palo Alto, CA: Electric Power Research Institute (EPRI).
- c. ESD-TR-86-278, Guidelines for Designing User Interface Software. Hanscom AFB, MA: Electronic Systems Division.
- d. NUREG-0700 Guidelines for Control Room Design Reviews. Washington, DC: Nuclear Regulatory Commission.
- e. Van Cott, H. P., & Human Engineering Guide to Equipment Design.
- f. Kinkade, R. G. (Eds.) Washington, DC: U.S. Govt Printing Office, 1972.

Standards:

- a. ANSI/HFS 100-1988 Visual Display Terminal Workstations.

Handbooks:

- a. ASHRAE Handbook of Fundamentals. New York: American Society of Heating, Refrigeration and Air Conditioning Engineers (ASHRAE).
- b. Helander, M. (Ed.) Handbook of Human-Computer Interaction. New York: North-Holland, 1990.
- c. Salvendy, G. (Ed.) Handbook of Human Factors. New York: Wiley, 1987.

G.8.2 Workplace Environment

The workplace and its environment are typically among the forgotten items in system design, their details often being left to the architect or the facility engineer. Substantial and costly adaptations may be necessary to adapt the facility to CCIS uses if optimum performance is to be attained. Among the relevant environmental factors that should be included in the CCIS's design considerations are the humidity and temperature, light, noise, and ventilation of the workplace. Each recommendation that follows is based on one or more of the documents cited in G.8.1, with particular emphasis on MIL-STD-1479, *Human Engineering Design Criteria for Military Systems, Equipment, and Facilities*.

Humidity and Temperature. For lightly clothed individuals with sedentary work in spaces with less than 45 feet per minute (ft/min) air flow, the ASHRAE recommends relative humidity between 20% and 60%, ambient temperature between 65°F to 85°F (18°C to 29.5°C), and less than 10°F difference between the head and floor levels. Temperatures between 70°F and 80°F (21°C and 27°C) are preferred by most people, with a relative humidity of 45% at 70°F (21°C). For comfort and work-force productivity, humidity should be decreased as temperature increases above 70°F, but not below about 15%. To minimize static electricity, humidity should be maintained at 40%, plus or minus 10%, and not fall below 20%.

Lighting. The illumination provided at the desk top and display should be sufficient to permit accurate reading. Since this differs from person to person, illumination level should be individually adjustable, and provide little or no glare. Acceptable illuminance ranges over an order of magnitude, from about 92 to 920 lx. On the average, day-time illumination (~240 lx average) should be greater than night (~180 lx average), and emergency lighting should not drop below the range of about 10 to 50 lx.

Noise. Ambient or background noise should not impair oral communication in the operating area, that is, it should not exceed the maximum level that permits person-to-person communication with normal or only slightly raised voice levels and normal hearing. Upper limits of permissible ambient noise are ~70 db(A) in routine-task areas (< 65 db are preferred), and ~55 db(A) where high levels of concentration are required.

Ventilation. Air flow into the conditioned workplace should be no less than 15 ft³/min per occupant, consisting of at least two-thirds fresh air, that is, outside air filtered to remove irritating or hazardous particles. NUREG-0700 recommends an air flow twice as great, that is, 30 ft³/min per occupant. Air flow should not produce a draft, and should not exceed 45 ft³/min at the head position of any workplace occupant.

G.8.3 Workplace Layout

It is likely that the future CCIS will be based primarily on *sitting workplaces*. Data and recommendations exist for standing and sitting/standing workplaces, as well, but considerations of these have not been included here. The importance of workstation design and workplace layout on user comfort, health, and productivity cannot be over stated. For example, [Kroemer 1988] reports that computer operators complain most frequently of health problems related to posture and vision, with musculo-skeletal pain and discomfort, eye strain and "fatigue" constituting at least half, and in some cases up to as much as 80% of all symptoms. At least some of the complaints appear to be related not only to each other (e.g., viewing difficulty, neck and lumbar strain, and arm and shoulder fatigue), but also to improperly designed or ill-arranged workstation furniture and layouts. Optimization of the workstation, with recognition of the range of sizes of personnel who may be assigned to use it, is or should be a fundamental goal of the WAM architecture --a goal that will result in increased comfort, safety, and productivity.

The recommendations that follow are based on one or more of the documents cited in G.8.1, with emphasis on three, namely, MIL-STD-1479, *Human Engineering Design Criteria for Military Systems, Equipment, and Facilities*; ESD-TR-86-278, *Guidelines for Designing User Interface Software*; and NUREG-0700, *Guidelines for Control Room Design Reviews*.

G.8.3.1 Adjustability and the Anthropometry of the User Population

The anthropometric dimensions of the projected user population, given the current policy of providing for a wide range of different physical sizes of U.S. personnel, mandates that the furniture or workstation components be adjustable to accommodate, for example, the range of sizes from the 5 percentile Oriental female to the 95 percentile Caucasian male. Adjustability also serves to permit the users to change their positions or work postures from time to time. The full range of sizes to be accommodated has been ascertained in most cases through anthropometric research sponsored by the Armstrong Laboratory at Wright-Patterson Air Force Base, Ohio. Some of the body dimensions of the U.S. civilian population have been published by [Kroemer 1981; 1988] in tabular form, and specifications for systems are given in MIL-STD-1472C.

G.8.3.2 Chair

The workstation chair should be designed to permit the user to maintain a posture with elbows, knees, and torso flexed $\geq 90^\circ$ (torso preferred range is 100° to 155°). Seat width and depth should be 17 to 20-inches (43 to 51-cm) and 15 to 18-inches (38 to 46-cm), respectively. Backrest width and height should be 12 to 14-inches (30 to 36-cm) and 6 to 9-inches (15 to 23-cm), respectively. Cushions should be at least 1-inch thick, and armrests (if provided) should be 2-inches (5-cm) wide, 8-inches (20-cm) long, and 7.5 to 11-inches (19 to 28-cm) above the sitting surface. Seat height should be adjustable from 14 to 22-inches (35 to 55-cm) above the floor. In particular:

- a. **Footrest.** A footrest, if provided, should be adjustable in at least 2-inches (5-cm) height increments, 18-inches (46-cm) below the high-positioned chair seat. If it is part of the chair, the footrest should be circular and of 18-inches (46-cm) diameter. Rectangular footrests should be about 12-inches (30-cm) deep by 16-inches (41-cm) wide.
- b. **Keyboard.** The height of the *keyboard base* --the worksurface on which the keyboard is placed—should range at least from 22 to 30-inches (56 to 77-cm) above the floor for sitting workplace. The height of the keyboard home row—the center letter row on a QWERTY keyboard -- should range at least from 26 to 30.5-inches (66 to 78-cm) above the floor. Both should be adjustable to suit different individuals.

- c. **Knee and Leg Room.** For the seated operator, knee clearance should range in depth from 18 to 20-inches (46 to 51-cm); leg clearances in depth and width should be 39-inches (100-cm) and 20-inches (51-cm), respectively.
- d. **Pointing Devices and Reach Envelope.** Pointing devices, including the finger for touch-sensitive screens, require a reach envelope of from about 25 to 35-inches (64 to 88-cm).
- e. **Position and Movement of the Head and Eyes.** The workstation layout should permit a seated operator to have both head and eye inclinations of between 16° and 22°, thereby supporting a viewing angle of between 32° and 44°. A provision for holding documents should be available to minimize the need for head movements while keying and viewing the display.

G.8.3.3 Screen Position, Orientation, and Viewing Distance

The viewing screen should be positioned approximately 13 to 30-inches (33 to 80 cm) from the user's eye point; most viewers prefer a range of from 18 to 24-inches (46 to 61 cm). The workstation's total vertical viewing angle should be within the upper limit of the visual field (75° above the horizontal line of sight) of the 5 percentile female, and a lower limit of 45° below the horizontal line of sight. The center of the outermost surface of the screen (less overlaying filters) should be within the range of 15° to 30° of the horizontal line of sight (below, for screen positions below the horizontal line of sight, or above for those positioned above). The screen should be angled so that the display surface is perpendicular a line from the user's eye. Screen orientation should be adjustable by the individual user. In particular:

- a. **Working Surface.** For a seated user, the level of the work surface used for writing, touch pad, or mouse, should be from 26 to 32-inches (66 to 81-cm) above the floor. The working surface should range in width and depth from about 24.5 to 39.5-inches (61 to 76.5-cm) and 16.5 to 25.5-inches (41 to 64-cm), respectively, with the wider and deeper surfaces preferred.
- b. **Workstation Hardcopy Printer.** Workstation hardcopy printers should conform to the recommendations of NUREG-0700.

G.8.4 Human Factors Engineering Principles

Issues related to the human factors engineering of elements not otherwise covered in this section should be resolved through application of the following nine principles of user-computer-interface design (reproduced from MIL-HDBK-761A, 30 September 1989 version, p. 27):

- a. **Acceptable Workload.** Workload should be within the capability limits of the user, and where possible, the user should direct system operation, and control the pace of transacting with the system.
- b. **Assurability.** The system should help assure data quality and transaction control by supporting the user in validating data, avoiding input errors, notifying the user of detected errors, and offering guidance in correcting errors.
- c. **Brevity.** User input and computer output should be brief and concise, and should reduce long and short term memory loads imposed on the user, and where possible, recognition rather than recall should be required of the user.
- d. **Compatibility.** User input should be compatible with computer output, and computer output should be compatible with human expectations, information assimilation capabilities, and information processing capabilities.
- e. **Consistency.** The system should provide a consistent interface environment and perform in a consistent, reliable, and predictable fashion.
- f. **Definition of Role.** The user should know what functions the user will perform and what functions the system will perform within dialogs.
- g. **Flexibility/Adaptability.** User input and computer output should depend on user experience, capability, expectation, and individual style, and should accommodate individual differences in style and abilities.
- h. **Feedback.** Immediate feedback should be provided the user concerning system status and user performance.
- i. **Simplicity.** User input and computer output should be formed into short, readily understandable structures.

G.9 OUTSTANDING ISSUES

There are several areas where current technology does not support adequate specification for the OSE profile. These are discussed below.

G.9.1 User Model

A model of expected user interaction is a critical part of the user interface architecture. The user model will drive all user interface design decisions such as selection of dialogue style and I/O devices. A formal model provides the basis for rigorous specification and design of reliable interfaces and provides a mechanism for using the common software engineering practice of abstraction to simplify the development process. It will help in automating the completeness and consistency checking of a user interface specification, and the development of a coherent and thorough test plan for verifying and validating a user interface implementation against its specification. It will also support determining the priority of such factors as ease of use and allow evaluation of the user's ability to use the system during early development activities. In addition to traditional task classification, CCIS users will benefit from a user model that accommodates classification by level of computer literacy, that is, as novice, intermediate, advanced, or expert. The preferred features differ for each level of expertise and a well-designed user interface should incorporate features applicable to all levels [Schneider 1979]. Experience level should be applied independently of specific functions, supporting a user at the novice level in one feature and advanced level in another, and should be dynamic allowing users to progress as they gain experience with a system.

A uniform interface protocol is a critical element of the user model. As more and more applications use a given protocol for interaction, the more benefit the user will gain from the associated set of skills in using it. There are no standards for specifying such a model. Since objects provide a natural representation of the elements of a user interface and a useful structuring mechanism for separating user interface and application code, an object oriented design approach is recommended for the CCIS.

G.9.2 User Interface Development

Interface design is an engineering problem that forces trade-offs among many factors. The pertinent questions are: What sort of interfaces should be supported? What constitutes a good set of programming abstractions for building such interfaces? How does a developer build a user interface given these abstractions? Currently there is a lack of top-down, quantitative principles which can guide this design. However, there are efforts in developing a quantitative assessment of how design trade-offs affect user satisfaction [JOPES COO 1986] and in developing quantitative tools for computing the operational parameters of user interfaces [JOPES PD 1988]. Meanwhile, methods that allow working qualitatively must be used (see, for example, [Schneider 1979; Grudin 1989; Linton 1989]).

User interface evaluation is critical. Typically evaluation of the HCI is limited to evaluation of the adequacy of information displays. Evaluation for the CCIS user interface must address all interfaces between the user and computer, not just the display interface and, to be effective in producing an improved system, it must begin earlier in the development process. The major issues are: What constitutes the interface? What constitutes evaluation? How can human-computer interaction be conducted early in the software development process? In particular, there is generally a gap between what can be easily and directly measured and the underlying concern being investigated, leading to additional issues relating to the accuracy of measurements and their representativeness of the real situation. The user interface architecture should be designed to help resolve these issues.

G.9.3 Aids For Cognitive Processing

The wealth of data generated by many current systems, and the diversity of tools provided to work with the data, can be highly confusing. Theory is being developed to understand how the ability of a human to work cooperatively with a computer in an effective manner results from an interplay among human mental processes, external computation, and memory aids. Once available, this theory should drive the development of new capabilities for communication and cooperation between humans and machines.

One area being investigated is that of intelligent interfaces. The underlying premise is that effective communication depends on shared cognitive task models embodying similar assumptions. Consequently, an intelligent interface uses qualities of directness and cognitive compatibility that may include dynamically adjusting the interface with respect to the

cognitive aspects of task composition. It incorporates knowledge about the application domain, the different tasks the user may perform, and the user himself (for example, his preferred interaction methods), typically in a highly interactive environment oriented to the problem at hand.

In practice, current intelligent interfaces are relatively primitive and take one of two forms. The first of these is exemplified by a human search intermediary in on-line information retrieval. Here the primary task is to understand a query given in terms of the user's model and translate the concepts of that query into task model terminology. The alternative form is that of a machine reasoning system which comprises user models, together with methods for representing task knowledge and inference tools for reasoning about these as the task progresses. Traditional AI techniques may be used for much of this capability.

G.9.3.1 Interaction History Aids

An interaction history facility allows a user to have access to past interactions and incorporate them in the current task to support reuse, recording and replaying a script, user recovery, navigation, external memory support, adaptive interfaces, and user modeling. Generally, these operations require little knowledge about tasks and can be application-independent, acting autonomously to relieve the user of the burden of mundane and repetitive tasks. This capability already exists in a limited form, for example, via macros for repeating a series of commands as a functional group and undo/redo commands. More realistically, basic history tools comprise four components: collection, presentation, selection/modification, and submission. The various design issues to be considered include:

- a. What should the history include? For example are both input and output objects required and should textual and non-text objects be recorded?
- b. What information can be obtained automatically and what must be provided with the user's own input? What techniques may be used to obtain the information?
- c. What portions of the history should be displayed? Should these be displayed permanently or on demand? What techniques and representational schemas can be used for reminders, external memory support, and cues for recognition and recall?

- d. What level of support and degree of functionality should be provided for the selection and modification of a selected history item?
- e. How should the history be organized, which items need to be integrated and kept separately? What support is needed to manage the history (e.g., accessing other histories, querying history, aging and discarding history)?

As yet, there is little knowledge about user interaction patterns and history usage characteristics, design concerns, and architectural concerns associated with history as a user support tool. Some architectural framework is needed to ensure that the history information from the relevant level is collected. Moreover, history concepts and functionality must pervade the system and transcend particular applications, yet be sufficiently flexible to support new concepts, uses, and capabilities.

APPENDIX H — WAM INFORMATION NEEDS

H.1 INTRODUCTION

This appendix provides background information and addresses the open requirements for WWMCCS. Discussed are the objective requirements of both a generic CCIS and the more specifically defined needs for the WAM Target Architecture in the 1995-1997 time frame. These operational requirements reflect the needs at the various military levels of responsibility and motivate the rationale for automated data processing (ADP) elements supporting the requirements. The ADP elements are divided into seven service areas, all of which are further explored in Appendices A through G. The connection of specific systems from these applications and categories of service in accordance with appropriate protocols and standards constitutes the basis of the WAM Target Architecture.

Section H.2 presents background information on the purpose, scope, and evolution of WWMCCS. Section H.3 identifies WWMCCS general requirements, responsibility levels, and functional needs. Section H.4 describes more specific requirements for the WAM target architecture, focusing on its seven service areas.

H.2 BACKGROUND

H.2.1 WWMCCS Purpose and Scope

DoD Directive 5100.30, December 2, 1971, paragraph III C, defines WWMCCS as the "system that provides the means for operational direction and technical administrative support involved in the function of command and control of U. S. Military Forces." WWMCCS is the central CCIS for the U. S. military and DoD-wide support. The Directive also specifies that WWMCCS comprises the following elements:

- a. The National Military Command System (NMCS)
- b. Command and control systems of the Unified and Specified Commands

- c. Related management/information systems of the Military Departments
- d. Command and control systems of the Headquarters of the Service component commands (the components of the Unified Commands)
- e. Command and control systems of DoD Agencies
- f. Interfaces with other Federal systems that have functions associated with the NMCS (White House Situation Room, State Department Operations Center, Central Intelligence Agency (CIA), Coast Guard Operations Center, etc.)

The directive establishes clear priorities as shown by the following extracts from paragraphs III and IV

The NMCS is the priority component of the WWMCCS designed to support the National Command Authorities in the exercise of their responsibilities. It also supports the Joint Chiefs of Staff in the exercise of their responsibilities.

The WWMCCS serves two functions, listed below in their order of priority and emphasis:

(1) Support of the NCA is the primary mission. The NMCS provides the means by which the President and the Secretary of Defense can receive warning and intelligence upon which accurate and timely decisions can be made; apply the resources of the Military Departments; and assign military missions and provide direction to the Unified and Specified Commands. The NMCS must be capable of providing information so that appropriate and timely responses may be selected and directed by the NCA and implemented. In addition, the NMCS supports the Joint Chiefs of Staff in carrying out their responsibilities.

(2) Support of the command and control systems of the Unified and Specified Commands and the WWMCCS related management/information systems of other DoD Components is the second mission. This function will be supported by the WWMCCS subordinate to and on the basis of non-interference with the primary mission.

Support of the NMCS will be the priority function of all primary and alternate command facilities.

H.2.2 Evolution of WWMCCS

The basic requirement for a WWMCCS originated with the 1958 amendments to the National Security Act of 1947. This Act establishes the concept of Unified and Specified

Commands. With this concept, the need for an effective command and control system, which would serve these commands and also bridge the unique service systems for higher authority requirements soon became apparent.

WWMCCS was officially established in October 1962 by DoD Directive 5100.30. Prior to that time individual Commanders in Chief (CINCs) and Services developed their own systems to enhance mission accomplishment. This resulted in different types of computers, incompatible software, inconsistent planning procedures, and inconsistent documentation. Planning data, for instance, stored in a particular command's computer systems, was readily available only to the organization that used that computer. Transferring information to another command's computer was mechanically difficult, frustrating, and time consuming. Moreover, the Unified and Specified Commands had, over time, developed different formats for storing the data to support their operation plans. Plans submitted by the combatant commands were difficult to analyze, review, and approve.

In 1966 the Secretary of Defense, recognizing the seriousness of the problems, directed the Joint Chiefs of Staff (JCS) to develop standardized joint planning procedures and a standardized ADP system that could be used with WWMCCS to support the new Joint Operation Planning System (JOPS). The new automated system was to perform the following:

- a. Foster common understanding by using standard procedures throughout the planning community.
- b. Give standard formats for operation plans that contain only the information necessary to understand and use the plans.
- c. Incorporate standard data files and common application programs in a system compatible with all users to allow the rapid flow of information.
- d. Permit the identification of shortfalls early in the planning process.
- e. Include a mechanism for plan refinement and review.
- f. Allow rapid conversion of the operation plan (OPLAN) into an operation order (OPORD) during a crisis.

Work began on the development of the new planning system in 1967 and the initial design of JOPS received formal JCS approval in 1970. The new guidance, planning procedures, and plan formats were printed in two volumes: Volume I (Unclassified) and Volume II (Classified).

By 1973, Honeywell 6000 computers had been installed to furnish the ADP support for the standardized procedures described in JOPS Volumes I and II. With the replacement of old computers many combat commanders lost use of their software, which was not compatible with the new equipment. Faced with the problem of losing previously developed ADP support for planning, the JCS directed rapid development of temporary computer programs until new software was introduced. The system was designated the JOPS Interim Software (JIS), and four Unified Commands were selected to design portions of it:

- a. U. S. Readiness Command designed a computer program to build and time-phase a force list, the Force Requirements Generator (FRG).
- b. U. S. Pacific Command developed a method of computing the support required to sustain a military force, the Movement Requirements Generator (MRG).
- c. U. S. Atlantic Command developed a program to simulate the strategic deployment of forces and support, the Transportation Feasibility Estimator (TFE).
- d. U. S. European Command designed the utility programs to allow the other major programs to communicate and produce a meaningful OPLAN database.

The JIS programs worked so well that they were adopted as the standard ADP system for joint operation planning. In 1975, JOPS Volume III was published, describing the computer support system. The JOPS ADP system, commonly called JOPS III, has undergone many updates since its original version.

By the end of the 1970s, 35 Honeywell 6000 computers were located in 24 locations, essentially at the headquarters of the major elements of the NMCS. Some of these headquarters found themselves with underused equipment and developed ways to use it for their own purposes. Thus, in some instances the definition of WWMCCS became blurred. However, the basic purpose and priorities for WWMCCS quoted at the outset were still in effect. A second concern during the 1970s was the centralization of databases in the Joint Staff with limited accessibility to the user of the bulk of the data.

Changes during the 1980s were primarily related to a modernization program entitled WWMCCS Information System (WIS). Its objectives were essentially to modernize WWMCCS software, replace or improve the hardware, and provide common user support. It focused on three organizational levels (National, Theater, and Supporting) and four operational phases (Deployment, Employment, Mobilization, and Sustainment). Priority was placed on Resource and Unit Monitoring and Conventional Planning and Execution in the

context of the type of information needed to make situation assessments and recommend appropriate action. A Joint Operations Planning and Execution System Required Operational Capability (JOPES ROC) was approved to guide these efforts [JOPES ROC 1983]. Some databases were decentralized, communications improved, and workstations proliferated. WIS, bogged down by multiple incompatible objectives, was terminated in 1990.

[This short history of WWMCCS was drawn from a number of references plus personal experiences of some of the authors [WWMCCS 1974, AFSC 1988].]

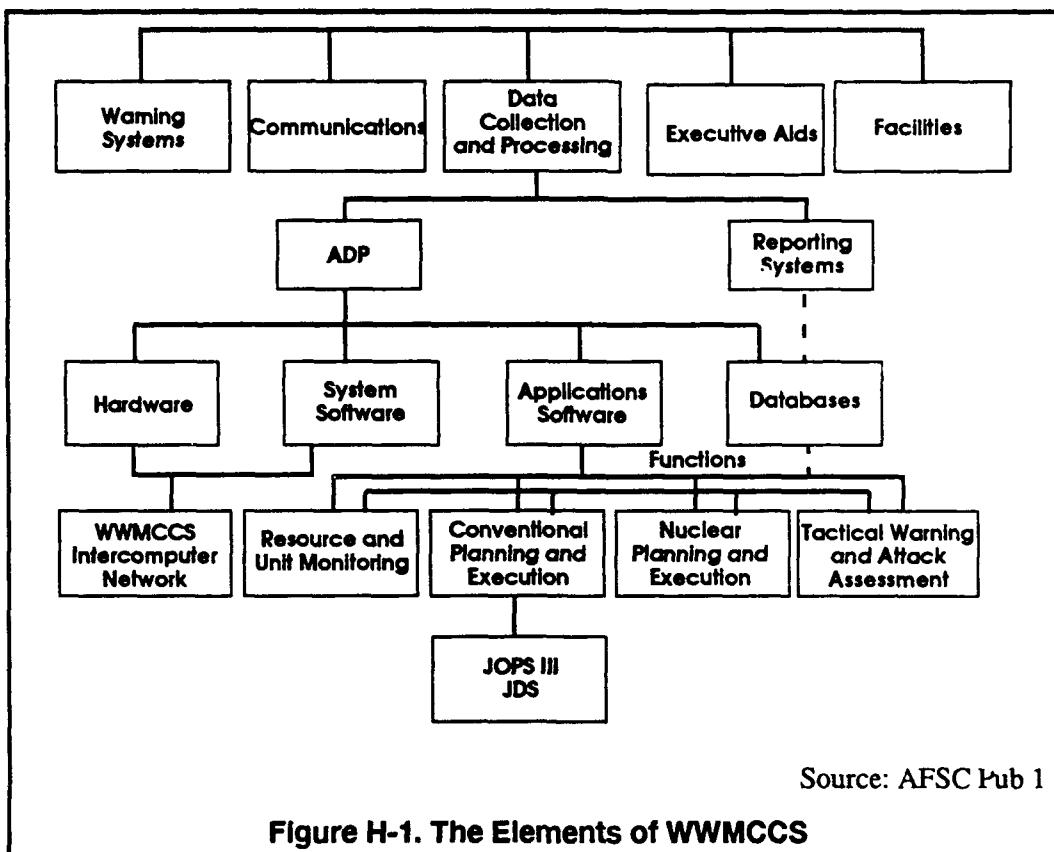
H.2.3 WWMCCS Today and Tomorrow

H.2.3.1 WWMCCS Today

The primary mission of WWMCCS is to support the national-level command and control function. On a noninterference basis, the system is available to support combatant commanders in their command and control responsibilities. As shown in Figure H-1, a conceptual view of WWMCCS includes five basic elements:

- a. Warning Systems: the tactical warning systems that notify operation command centers of the occurrence of a threatening event.
- b. WWMCCS Communications: the general and special-purpose communications capabilities to convey information, hold conferences, and issue orders.
- c. Data Collection and Processing: the collection and handling of data to support information requirements of WWMCCS.
- d. Executive Aids: the WWMCCS-related documents, procedures, reporting structure, and system interaction that permit the user to connect with the system, enter data, and receive output records, forms, and displays.
- e. WWMCCS Command Facilities: the primary or alternate command centers.

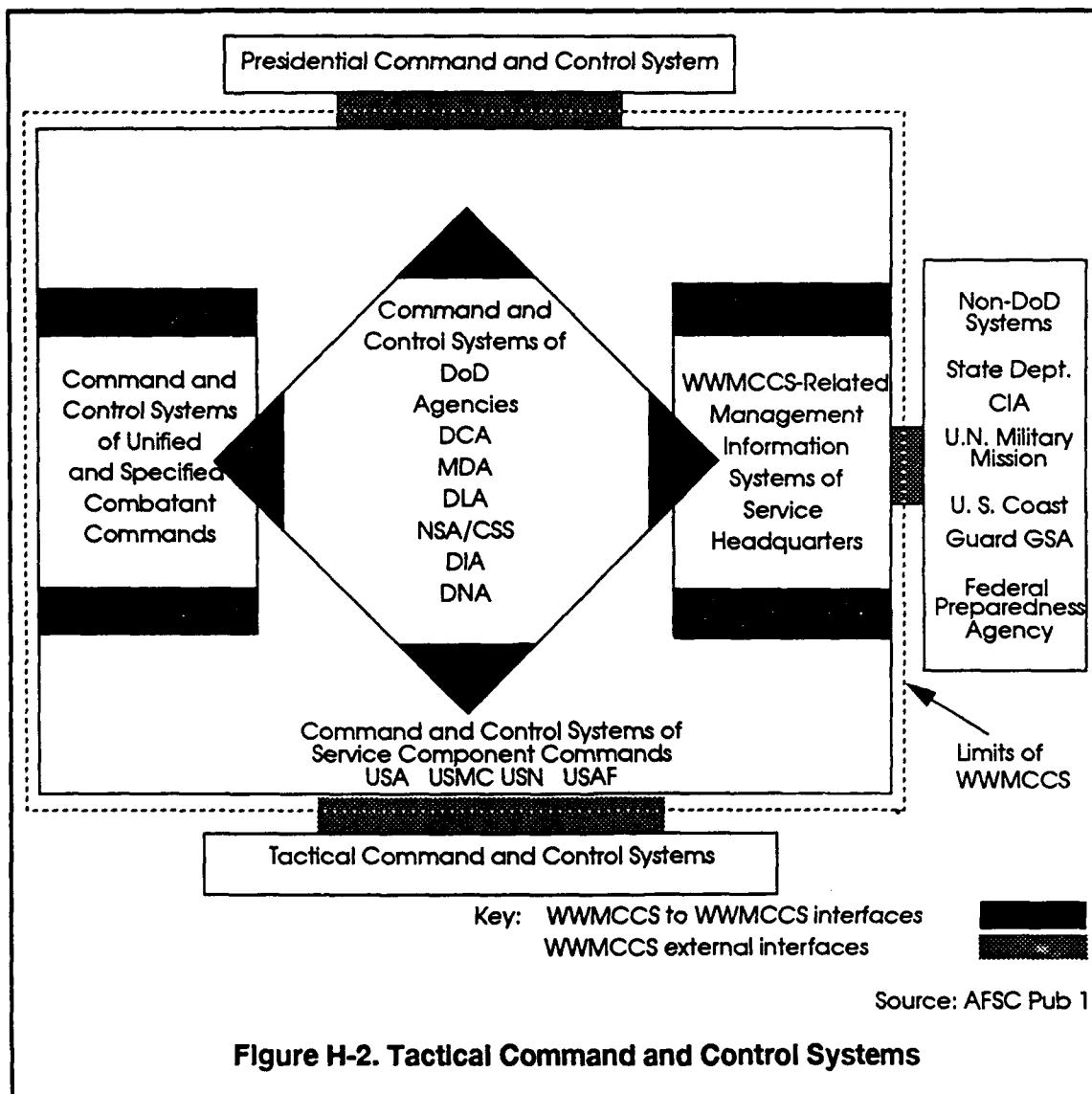
Each of these extends through the various levels of command and control. The operation of the WWMCCS elements together forms a worldwide information system. The Data Collection and Processing element has been expanded in Figure H-1 to show additional detail and to illustrate the relationship of JOPS III and the Joint Deployment System (JDS) ADP to the overall system.



The basic function of WWMCCS is to develop and transfer information. The information flow is enhanced both by formalized reporting systems defined in the JCS Pub 1-03 series and by standard systems connected together in a network of reporting systems and databases. Four basic functional areas are supported: Resource and Unit Monitoring (RUM), Conventional Planning and Execution (CPE), Nuclear Planning and Execution (NPE), and Tactical Warning/Attack Assessment (TW/AA). Users of the current WWMCCS interact with it through visual information projection (VIP) terminals or WIS workstations such as the IBM-PC/XT. These devices are connected to one of the many Honeywell 6000 computers that have since 1973 been the standard ADP support for joint operation planning and execution. These computers have been substantially upgraded and are now called the Distributed Processing System-8 (DPS-8).

WWMCCS is not a single system. It is a system of systems that range from the national to the theater level. Some component systems are WWMCCS unique, but most are designed, developed, purchased, and used to satisfy the command and control requirements of the Services or commands that normally use them. WWMCCS is not a closed system.

Figure H-2 illustrates interfaces with non-WWMCCS systems, non-DoD agency systems,



and tactical command and control systems that support military forces.

The WWMCCS Intercomputer Network (WIN) permits users to communicate, to review and update data at other WWMCCS locations, and to transfer data between computers. The land line and satellite connections permit real-time Top Secret communications. These capabilities are described briefly in the following subparagraphs:

- a. Telecommunication Network (TELNET) is used to establish remote access to computer resources of another remote host in the network; that is, with proper

- permissions users can log on to a WWMCCS remote host computer site as if the terminal were connected to their site.
- b. File Transfer Service (FTS) is used to exchange large volumes of data; for example, entire TPFDD files can be passed between members of the Joint Deployment Community (JDC).
 - c. The WIN Teleconference (TLCF) permits up to 80 interconnected WWMCCS terminals to confer and exchange textual information simultaneously.

[This section is in part extracted from various portions of AFSC 1988.]

H.2.4 What WWMCCS Must Support

WWMCCS must be capable of supporting the military forces over a spectrum of options ranging from support to civilian agencies (i.e., drug interdiction) to general nuclear war. Further, it must support each echelon of command from the NCA to the Joint Task Force (JTF) level. The primary challenge is to provide timely and accurate information for any given situation in the degree of detail and format that is useful to support decision making at each echelon involved.

H.2.4.1 Support to NCA/JCS

In general, the National Command Authorities (NCA) require detailed information for developing crises and small contingencies but only aggregated information for major operations. Some of these differences and the rationale are contained in Table H-1 on page 9 [IDA 1982]. A report prepared for the DoD and the State Department in 1980 suggested a realistic scenario of a major crisis preceding a war in Europe and illustrates the types of questions asked by the NCA in a major crisis [Rand 1980]. As intelligence indicators were passed to the NCA, their questions all involved assessment, rather than details of the intelligence reports:

- a. Is there a crisis coming?
- b. What is the meaning and significance of the intelligence indicators?
- c. What options are open to the United States?
- d. What is involved in the U. S. response?

Table H-1. Comparison Between Small Contingencies and Large Operations

Small Contingencies	Major Operations
Initiating event may be uncertain or ambiguous	Initiation clear
Sudden initiation more likely	Gradual development more likely
Enemy objectives may be unclear	Enemy objectives clear
NCA defines objective at the time	U. S. objective essentially predetermined to win
NCA explores many options	U. S. military options essentially predetermined
NCA demands preevaluation of outcomes	Outcome preevaluation of lesser priority
NCA demands precise control of operations	Precise NCA control of operations impossible
Normally closely held, at least in beginning	Security an issue but broader interfaces possible
Minor forces involved	Major forces involved
Sustainability not usually a critical issue	Sustainability a key issue
Military information for NCA is centered in the JCS	Military information for NCA must be drawn from entire defense community
Essentially political in nature	Essentially military in nature
Plans amendable to last minute revisions	Scale and complexity of plans make last minute changes difficult

Only after the NCA were convinced that the crisis was real would they begin to pay attention to details of the U. S. response.

The NCA look to the NMCS, which is part of WWMCCS, and particularly to the JCS, to provide information, options, and recommendations concerning the use of military power. Table H-2 on page 10 lists the key responsibilities of the JCS as embedded in statute and implementing documentation. The analysis contained in IDA's Report R-266 [IDA 1982] concluded that these responsibilities were closely interrelated and that one—(No. 3) preparing estimates of the situation—was central to the others. Three of the others (Nos. 1, 2, & 9) are largely information inputs to the estimate, while the completed estimate provides much of the information necessary to the fulfillment of the other six responsibilities. Therefore, the highest priority for WWMCCS is to provide the information needed to fulfill these responsibilities in a satisfactory and timely manner.

Table H-2. Key Responsibilities of the JCS

Responsibilities	Derivation				
	Title 10 U. S. Code		DoD Directives		JCS Pub 4
	Explicit	Implicit	Explicit	Implicit	Explicit
1. Monitor worldwide activities to identify potential crisis areas.		X		X	X
2. Review existing military plans applicable to potential crisis areas for adequacy, feasibility, & suitability; develop plans if necessary.		X	X		X
3. Prepare estimates of the situation, emphasizing alternative courses of action/options and related risks.		X		X	X
4. Recommend options and provide information to the NCA.		X		X	X
5. Provide strategic direction to armed forces.	X		X		X
6. Assign logistic responsibilities to the military Services and the Defense Logistics Agency.		X	X		X
7. Adjust priorities & allocate forces/resources in accordance with the situation.		X		X	X
8. Supervise implementation of integrated plans for military mobilization.		X	X		X
9. Provide joint intelligence for use within the DoD.		X	X		X
10. Issue operational directives.		X	X		X

H.2.4.2 Support to CINCs

The primary sources of the information to accomplish the above are the Unified Commands, their component commands, the Services, and the specified commands. The five combatant commanders have geographic area responsibilities, and are responsible for all joint operations within their designated areas: U. S. Atlantic Command (USLANTCOM), U. S. European Command (USEUCOM), U. S. Central Command (USCENTCOM), Pacific Command (USPACOM), and Southern Command (USSOUTHCOM). The CINCs of the remaining combatant commands have worldwide functional responsibilities not bounded by any single area of operations: U. S. Space Command (USSPACECOM), U. S. Transportation Command (USTRANSCOM), U. S. Special Operations Command (USSO-

COM), and the specified combatant commands, Strategic Air Command (SAC), and Forces Command (FORSCOM).

Some of the commands serve in more than one capacity. For example, USFORSCOM serves as the Army component of USLANTCOM. As a specified command it also provides a general reserve of combat-ready ground forces to reinforce other CINCs and is responsible for readiness and related deployment planning for those forces. Furthermore, it performs the land defense of the continental United States (CONUS) and Canada, supports civil defense efforts and the Joint Key Asset Protection Plan. FORSCOM is also a major command of the U. S. Army.

While five geographical area Unified Commands have the same general mission for separate geographical areas, the assigned geography itself creates different environments within which they must operate. These differences include such things as potential enemy capabilities, terrain, in-theater forces, in-place infrastructure, distance from CONUS, alliances, host nation support facilities, capability of allied forces, etc., all of which have an influence on a WWMCCS architecture. When the functional Unified and Specified Commands and the DoD agencies are added, it is obvious that a broad requirement of the system is to ensure that the senior echelons are not overwhelmed with detailed information, yet details are available when the NCA need it.

H.2.4.3 Support to Deliberate Planning and Crisis Management

All military operations require planning—such planning may be conducted in a few minutes or be the product of years of effort. The joint planning process is designed to be a logical procedure that results in sound decisions. In peacetime plans provide the basis for program and budget decisions and a common starting point in war should they be implemented. In this regard, the late General Creighton W. Abrams defined a war plan as “a common point of departure for change once the action starts.” The process is iterative.

Since the amount of time available significantly influences the planning process, two different methods of planning are used within the Unified Command system.

- a. Deliberate or Peacetime Planning is the process used when time permits the total participation of the commanders and staffs of the Joint Deployment Community (JDC). The development of the plan, coordination among supporting

commanders, reviews by the Joint Staff, and communications between the members of the JDC take many months; preparation of a large plan may not be completed before the next two-year cycle begins. JOPS is used for this planning process.

b. Time-Sensitive or Crisis Action Planning is conducted during emergencies. The overall process parallels that of deliberate planning, but is a more flexible system that responds to the demands of changing events. The procedures allow for a logical, rapid flow of information, timely preparation of executable sources of action, and communication of the decision of the NCA to the combatant commander. The ADP support for time-sensitive planning is supplied by the Joint Deployment System (JDS); the demands for plan execution and monitoring are quite different from those of deliberate planning and its support with JOPS ADP.

Figure H-3 summarizes the steps in the two procedures. The planning cycle for the deliberate planning process begins with the publication of the principle task-assigning document, the biennial Joint Strategic Capabilities Plan (JSCP), and ends in the last year of the JSCP. The approved OPLANs prepared as directed by the JSCP are considered effective until superseded. Crisis Action Planning (CAP) begins with some incident or development of a situation that might precipitate employment of military forces. If there is an existing OPLAN or concept plan (CONPLAN) that can be applied to the situation, then it is used or adjusted to fit the specific situation. Lacking either, the process starts with examining alternatives to address the crisis. Depending on the nature of the crisis, forces to be employed may come from one or more of the combatant commands or CONUS and in most instances will require movement. Therefore, the JDS plays an important role.

Table H-3 on page 14 compares crisis action and deliberate planning procedures. The principal differences are in the time available for planning and the degree of involvement of the full set of planners. Deliberate planning is a lengthier process, involving the full planning community and not reacting to a specific provocation or incident demanding a rapid response. As the table indicates, up to two years may be spent developing the plan. Crisis action planning, on the other hand, is usually done under urgent time and political constraints, may involve very few planners in a close-hold operation, and can culminate in plan execution when so authorized by the NCA.

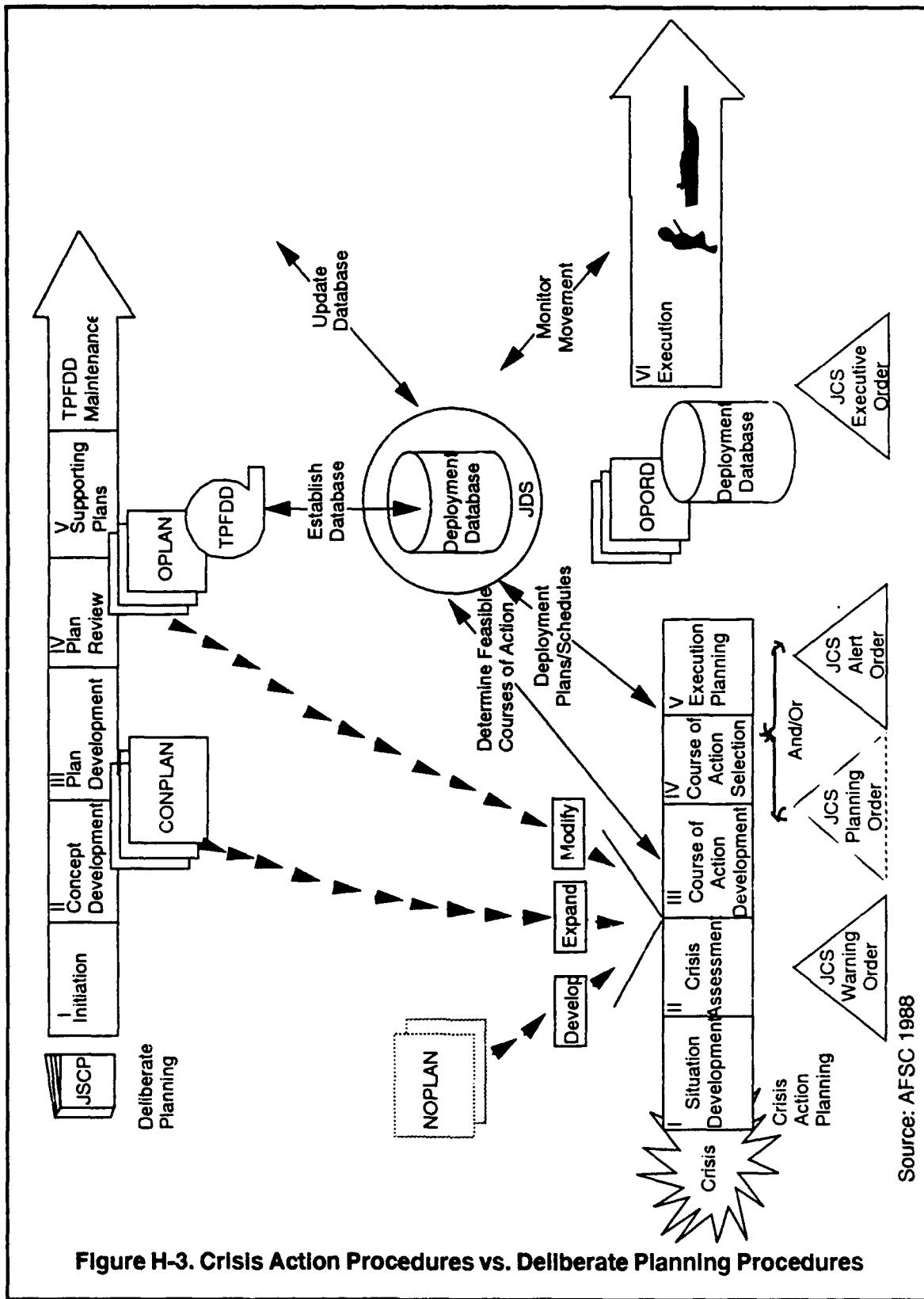


Table H-3. Crisis Action vs. Deliberate Planning

Attribute	Crisis Action	Deliberate
Time available	Hours or days	18-24 months
JDC involvement	Possibly very limited	Participates fully
Phases	6 phases from situation development to execution	5 phases from initiation to supporting plans
Document assigning task	Warning Order	JSCP
Forces for planning	Allocated in warning, planning, alert, or execute order	Apportioned by JSCP
Early planning guidance	Warning order from CJCS CINC's evaluation request	Planning directive issued by INC after planning guidance step of concept development
Commander's estimate	Communicates recommendations of CINC to CJCS-NCA	Communicates the CINC's decision to staff and subordinate commanders
Decision of COA	NCA decides COA	CINC decides COA, CJCS reviews
Execution document	Execute order	When implemented a plan is converted to an OPORD and executed with an execute order
Products	OPORD with supporting plans	OPORDs, OPLAN or CON-PLAN
ADP support	JDS procedures manual	JOPS Volume III

H.2.5 Planned WWMCCS Improvements

The current major effort to improve WWMCCS capabilities is the WAM Program. Spe-

cific improvements are programmed in a succession of versions for completion in 1995. From an operational requirements perspective these enhancements are currently centered in JOPES. Presently, there are two separate systems for war planning and execution, JOPS and JDS. Unfortunately, JOPS and JDS grew through a series of compromises designed to preserve the best of the numerous systems that had been used throughout the joint planning community. The result has been the development of systems that are not capable of performing all desired functions.

Recognition of these shortcomings resulted in the approved JOPES ROC, dated 5 July 1983. Since then the JOPES Concept of Operations was published in 1986 [JOPES COO 1986] and a JOPES Procedures Description in 1988 [JOPES PD 1988]. JOPES is visualized to be an integrated part of WWMCCS and is planned to satisfy the information needs of senior decisionmakers in conducting joint planning and operations. JOPES will be used to monitor, plan, and execute mobilization, deployment, employment, and sustainment activities at the national, theater, and supporting responsibility levels, both in peace and war.

Collateral support to the Joint Strategic Planning System (JSPS) and the Planning, Programming, and Budgeting System (PPBS) in identifying and analyzing force requirements and capabilities is also planned for WAM. The primary emphasis is on procedures, supported by modern ADP and communications systems, to replace the time-consuming machinery of current systems. JOPES will not cause events to happen during execution, but will give senior-level decisionmakers the tools to monitor, analyze, and control events during execution.

The JOPES concept can be visualized as seven interrelated functions: monitoring, threat identification and assessment, strategy determination, course of action development, execution planning, implementation, and simulation and analysis. The JOPES functions encompass the planning phases of the current deliberate and time-sensitive planning systems outlined in JOPS Volumes I and IV. A principal goal of JOPES is to develop one set of procedures for both deliberate and time-sensitive planning, differentiated primarily by the length of the planning cycles; JOPES will work toward 45 days for deliberate planning and 2 to 3 days for crisis planning.

JOPES development is being handled in an evolutionary manner. About 25 percent of the 80 JOPES support elements proposed under the now defunct WIS program have been included in Increment 1, for which 11 versions are currently planned (JOPES is to be fully operational in 1995). The purpose of Increment 1 is to establish a baseline for a single, inte-

grated planning system. The first increment of joint, automated capabilities is to be built primarily on current JOPS, JDS, logistics systems, and selected portions of both Status of Resources and Training System (SORTS) and the NMCC Information for Decisionmakers System (NIDMS). As further interfaces are established with Service-, command- and agency-unique systems, less reliance on joint mission applications will be needed for determining requirements and assessing resources.

H.3 WWMCCS REQUIREMENTS

H.3.1 General

This section summarizes WWMCCS operational requirements and provides a guide for estimating the software and computer needs. Thus, this section provides the connection between the functional requirements for all C² centers and the more detailed technical appendices to this report. A comparison of the requirements and the features provided within each of the seven service areas of the target architecture is provided in Section H-4.

WWMCCS support to the NCA and deployed forces has grown significantly in the nineteen years since DoD Directive 5100.30 was approved. For example, since 1981 the Marine Corps has increased the number of WWMCCS remote terminals from 3 to 26. This will provide a WWMCCS capability for Headquarters Marine Corp, Fleet Marine Force Atlantic and Pacific Headquarters, and Marine Air Ground Task Forces (MAGTFs) down to brigade level. Additionally, within USCINCLANT there are four Deployable Automated Data Terminal Response Teams (DART). These DARTs consist of both personnel and hardware to operate a deployed WWMCCS site. There are also four "tactical user ports" allocated for DART use through the CINCLANT host computer. USFORSCOM also plans for a WWMCCS Entry System (WES) of about 220 terminals located in 150 different sites. Today's WWMCCS is on the verge of further expanding its capabilities as a worldwide command and control system, a system intended to provide automated means for the operational support involved in the command and control of U. S. Military Forces. The WAM program is at the heart of the present phase of this expansion and is a prelude to any future CCIS.

The functioning of a WWMCCS, in support of these missions, requires interaction among many levels of responsibilities and an increasingly wide spectrum of interfaces for

information, coordination, execution, and feedback. One key to the future success of WWMCCS will be its interoperability and responsiveness. Another key will be the recognition that no worldwide C² system, with a potential for directing and coordinating military operations in several theaters and at different levels of conflict, could possess in one database all the required relevant information for planning and decision making. Hence, a continuing stream of updating information involving several databases is needed. With aggregation and interaction extending from the national level to the tactical level, between the supported forces and the supporting commands and agencies, and the NCA, the data management task is formidable.

H.3.2 Responsibility Level

Command and control responsibilities reside in three levels: (1) the National Level, (2) the Supported Theater Level, and (3) the Supporting Levels. The composition of each level is given in Table H-4 on page 17.

Table H-4. Composition of Responsibility Level

Level I National
National Command Authorities (NCA)
National Security Command (NSC)
Chairman, Joint Chiefs of Staff
Joint Staff
Service Headquarters
Level II Supported Theater
Supported Commander
Supported Components
Level III Supporting
Supporting Unified Commander
Supporting Unified Components
Major Service Management Commands,
Reserve Component Organizations, Logistic
Commands, Personnel Centers, Communications
Commands
Defense Agencies (OSD, DLA, DIA, DMA, DNA,
DCA, and NSC)
Federal Agencies (FEMA, Commerce, Labor
Transportation, Health, Human Services,
Selective Service System, General Services
Administration)
Allied Commands

Source: [JOPES ROC 1983]

The primary decision-making responsibility for developing strategy, deliberate and crisis planning, selecting courses of action (COAs), for resource allocation and for oversight of deployment planning resides at Level I. The Level II responsibilities of the executing and supported Unified Command forces involve developing the detailed operation order, formulating needs for force deployment, and execution planning. Level III is instrumental in proposing and subsequently providing transportation, logistic support, and additional combat and support forces. Considerable information exchange must take place laterally and vertically between and across all levels in support of the initial decision by the NCA to respond through a selected COA.

H.3.3 Functional Needs

The nature and amount of data required by any particular level within the WWMCCS varies with the circumstances. Level I requires only summary data for extensive conflict but may need specific details in a particular crisis. The CAP outlined in JCS Pub 5-02.4 (previously JOPES III) serves as an indicator of the various functions performed outside peacetime planning. Since these functions tend to be crucial to rapid decision making, their satisfaction is the most demanding requirement of a CCIS.

The steps shown in H-4 are composed of six stages or phases, and show the interactions required in responding to a crisis by developing COAs, selecting one COA, and subsequently carrying it out.

Satisfaction of the activities noted in the CAP is required to provide Defense-wide intelligence information and communications support for force management, decision making, and technical support of DoD operations in peace, crisis, and during all phases of conflict. An information system for all command centers must concurrently support both deliberate planning during peace and execution planning in crisis. The high-level functions to be supported in all cases are (1) planning, (2) decision making, (3) mobilization, (4) deployment, (5) employment, and (6) sustainment.

Basic and fundamental mission needs to support C² functions were initially set out in the JOPES ROC [1983, 15] and are summarized as follows:

- a. Facilitate development, review, and assessment of OPLANs in single and multiple theater environments.

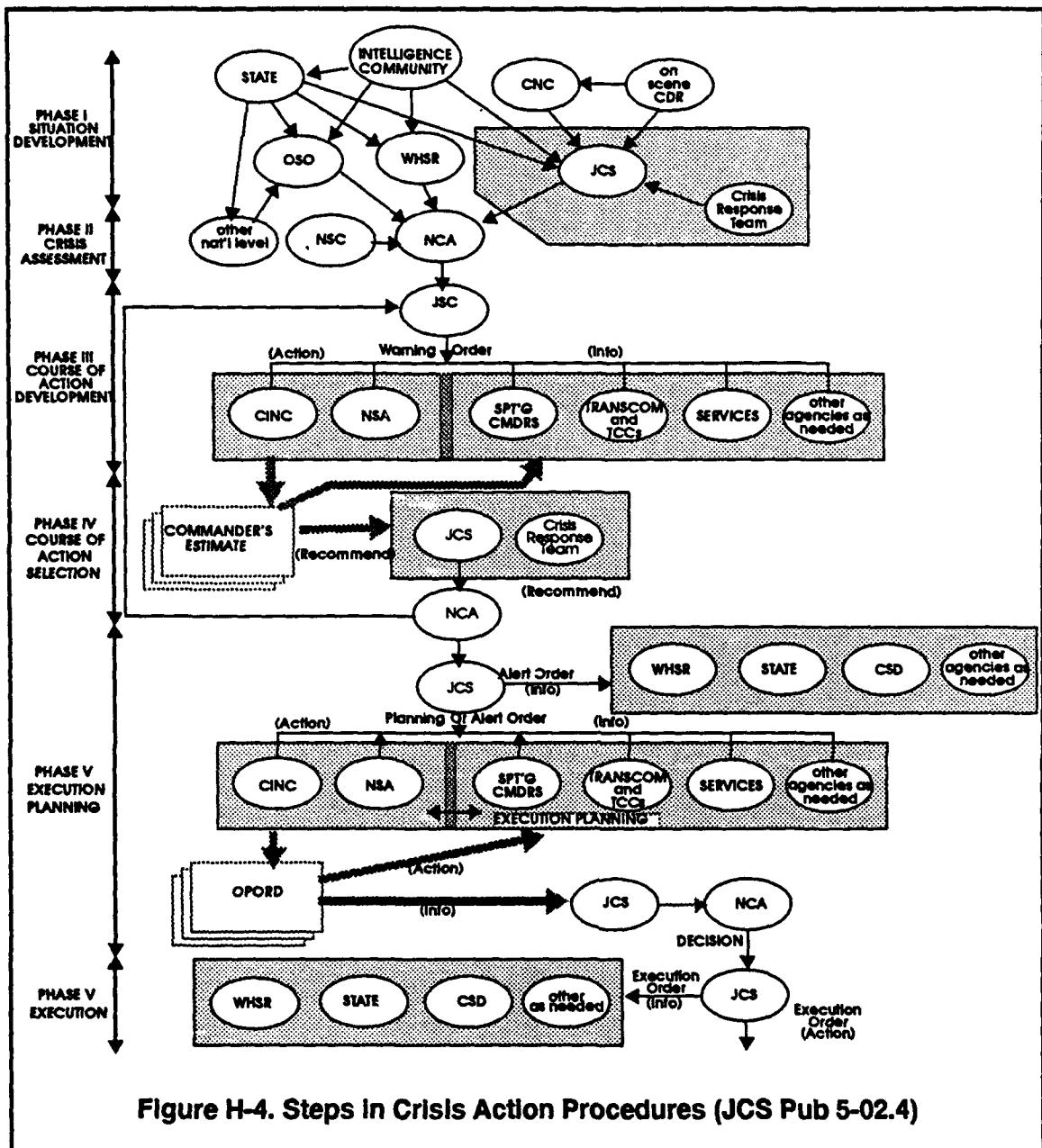


Figure H-4. Steps in Crisis Action Procedures (JCS Pub 5-02.4)

- Assist in rapidly developing executable COAs in planned and no-plan situations.
- Assist in developing and evaluating proposed COAs in a multi-theater environment.

- d. Facilitate integration and prioritizing combat, combat support, combat service support forces, sustainment, resupply, personnel, transportation, and command, control, and communications (C³) resources in a single or multi-theater environment.
- e. Facilitate preparation, development, and promulgation of warning, alert, execution, mobilization and deployment orders by providing relevant data on mobilization, forces, readiness, logistics, personnel, transportation, and C³ resources.
- f. Provide improved automated dynamic support for manipulation of plans and schedules to allow efficient execution of decisions to initiate, redirect, modify or stop deployments as the situation dictates.
- g. Provide capabilities to monitor readiness, mobilization, deployment, employment, and sustainment of forces and resources and facilitate preparation of progress reports; identify problems and, when required, rapidly develop alternative solutions.
- h. Facilitate planning and execution of time-sensitive close-hold operations.
- i. Provide the redundancy required to function in a communications and ADP-degraded conditions.

These mission needs for JOPES were later addressed by the Joint Requirements Oversight Council (JROC) on 21 September 1989. The JROC validated these requirements and incorporated them into a Mission Need Statement for JOPES [JOPES MNS 1989]. A summary of these requirements for planning, decision making and execution follows.

- a. Posture national civilian and military leaders to propose, select, and implement preferred courses of action to achieve specific political and military objectives.
- b. Provide timely and accurate decision-making tools that keep pace with crisis situations, yet will support the detailed requirements of the deliberate planning process.
- c. Provide senior leaders and staff officers with timely, accurate, complete, and properly aggregated information that serves the decision-making process.
- d. Provide for continuous monitoring of the global situation to include the status of U. S. forces and resources, threat indications and warning, and the capabilities of potential adversaries.

- e. Provide the analytic tools required to support rapid development, evaluation and selection of strategic options and military courses of action in single and multi-theater scenarios.
- f. Support execution planning requirements for the development of OPLANS within forty-five days of concept approval and OPORDS within three days of NCA course of action selection.
- g. Facilitate the evaluation of plans, including their underlying assumptions and probable implications, in order to assess capabilities, identify shortfalls, and decide resource prioritizing.
- h. Assist commanders to start, stop, or redirect military operations effectively in response to changes in guidance, resources, or threat.
- i. Facilitate mobilization, deployment, employment, and sustainment planning and execution.
- j. Integrate existing systems for planning, decision making, and execution within a single architecture defined by established standards and policies.
- k. Interface with existing and planned Service and Defense Agency databases and information systems.
- l. Provide policies and procedures that are similar, if not identical, in peacetime, crisis situations, and war.
- m. Exploit technological advances in information systems and communications.
- n. Safeguard information from unauthorized access, manipulation, or retrieval.

H.4 WAM TARGET ARCHITECTURE 1995-1997

H.4.1 Information Needs

As noted earlier, different responsibility levels within the CCIS require the same categories of information but not necessarily the same level of detail. Moreover, the level of detail demanded by Level I depends, in part, on the political sensitivities surrounding the planning and execution phases of the CAP. Nonetheless, it is noted that even if the highest responsibility levels demand extensive details rather than summaries during a sensitive operation, the lower levels must have even more detailed information to support the infor-

mation needs of the NCA. Thus the needs for differing levels of detail distinguish the requirements of the three levels. Some of the differences in the level of aggregation of information needed at different responsibility levels can be appreciated from an examination of the three tables that appear at the end of this appendix. These tables summarize information requirements contained in the JOPES ROC Annex B Tab B, dated 1983.

The differences in information needed in *plans* can be clearly seen in these tables. Level I planners need summaries with some indexing to the type of plan, the region covered, and certain keywords for identification. Table H-5 on page 34 shows this requirement. Level II planners, responsible for executing the operation, need greater detail. As Table H-6 on page 36 shows, the plan information at Level II is specific for each geographical region of the operating area and includes either aggregated or possibly even detailed force lists. Table H-7 on page 38 shows that CINCs or other organizations at Level III who support the operations in the plan must have very detailed information if involved directly and aggregate information if involved incidentally.

The level of detail varies across command levels although there is a need for all commands to use the same information when planning or conducting operations. If it is possible for a CCIS to store only a single data element for each fact and provide summarization and aggregation in a way that is transparent to the users, then the information needs for most CCIS users could be met from a single data source.

In the following sections, these information needs are expanded and grouped according to the seven target architecture service areas.

H.4.2 Target Profile Requirements by Service Area

This section addresses the technical services and associated requirements necessary to meet those broad operational mission needs and tasks highlighted earlier. These technical services are key to the proper functioning of all WWMCCS applications.

To meet those requirements, the target architecture incorporates seven service categories: data exchange service, data management service, network service (including communications and connectivity), operating system service, programming service, security service, and user interface service. Approved documents [NIS ROC 1983; AMH ROC; JOPES ROC 1983; JOPES MNS 1989; and CINC C² Master Plans] are sources for the

requirements summarized in this paper. (The JOPES ROC, page 65, contains a listing of generic technical support capabilities required for JOPES.) The particular source is indicated in brackets after each requirement. Where no source is explicitly indicated, the requirement should be viewed as a technical opinion of the feature needed to support one or more stated requirements.

H.4.2.1 Data Exchange

- a. Database management system (DBMS) to accommodate distributed databases with access to various databases transparent to users [NIS ROC 1983, 25, 48; JCS PUB 6-03.10, Vol II Annex B, B-5]
- b. Simultaneously transmit graphics and data to other WWMCCS sites [NIS ROC, 14]
- c. Send and receive graphics for large screens and television [NIS ROC 1983, 14]
- d. Information exchanges between [JOPES ROC 1983, 20, 21; NIS ROC 1983, 25]
 - 1) All Joint Staff Directorates
 - 2) Service Headquarters
- e. Defense Agencies, the CINCs, the Intelligence Community (CIA and NSA), the State Department, the White House, FEMA, other federal government agencies, and Intelligence systems with multilevel security, such as DODIIS.
- f. Protocols will provide high speed atomic information exchange between nodes [JOPES ROC 1983, 13]
- g. Data transmission verification and retrieval [JOPES ROC 1983, 63, 65]]
- h. Interactive processing [JOPES ROC 1983, 90, 105]
- i. High-density quick storage medium such as video disc capable of displaying reference documents and maps in a real-time mode [NIS ROC 1983, 9]
- j. Assured interoperability between NMCS and major operational command centers [NIS ROC 1983, 25]
- k. Capability to display and transfer a working color map between two or more headquarters [NIS ROC 1983, 8]
- l. Teleconferences [NIS ROC 1983, 5]

- m. Computer conferences [JCS Pub 6-03.10 Vol II Annex B, B-5]
- n. Data processing element exchange using voice, video, graphics and data systems [NIS ROC 1983, 5]
- o. Graphics support through a TEMPEST qualified color graphics terminal will be required for data presentation and analysis that interfaces electronically with finished graphics processes [NIS ROC 1983, 5; JCS Pub 6-03.10 Vol II Annex B, B-7]
- p. Voice [NIS ROC 1983, 5]
- q. Video capabilities to simultaneously display classified command and control related data at a number of locations within an organization or facility via a television set or cathode ray tube repeater [NIS ROC 1983, 5; JCS PUB 6-03.10 Vol II Annex B, B-8]
- r. Text messages [NIS ROC 1983, 5]
- s. Data pattern messages [NIS ROC 1983, 5]
- t. Computer graphics conferencing [JOPES ROC 1983, 106; NIS ROC 1983, 5]

H.4.2.2 Data Management

- a. Congruent database structures [NIS ROC 1983, 1, 25]
- b. Distributed and widely separated source databases [JOPES ROC 1983, 13; JCS Pub 6-03.10 Vol II Annex B, B-5]
- c. Component database structures and reporting systems [NIS ROC 1983, 15]
- d. DBMS should permit the data field to vary in length from record to record in file. The use of standard DBMS techniques and software will reduce application development time and cost, increase flexibility and adaptability of applications, and contribute to interoperability and interfaces among various management information systems within the WWMCCS. The DBMS will support applications interface to various database structures. [NIS ROC 1983, 15; JCS Pub 6-03.10 Vol II Annex B, B-5; JOPES ROC, 104]
- e. System will require near-real time automated capabilities to receipt, store, process, display, and integrate all environmental data in support of air, land, and sea operations, surveillance and other areas as necessary [NIS ROC 1983, 12]

- f. USCENTCOM requires environmental data to support air, sea, and land operations information; it must be near real time to be of maximum value to decision makers and support current operations [USCENTCOM C3 Master Plan June 1989, 2-15]
- g. Near real-time Joint Reconnaissance Center management desired processing speed to change from one to two days to five to ten minutes [NIS ROC 1983, 11]
- h. High-density quick storage medium such as video disc capable of displaying reference documents and maps in a real-time mode [NIS ROC 1983, 9]
- i. Capability for a more advanced automated filing system for large volumes of dynamic relatively discrete data [NIS ROC 1983, 7]
- j. System accounting and performance monitoring [JOPES ROC 1983, 105]
- k. Generic support capabilities for data formatting, collection, analysis, storage, and retrieval [JOPES ROC 1983, 103]
- l. Event-driven processing to satisfy near real-time information requirements [JOPES ROC 1983, 103]
- m. Ensure the access to distributed databases for maintenance and selective update on a number of different computer systems at different locations featuring multiple-source location and correlation without user prompting [JOPES ROC 1983, 103, 117]
- n. State-of-the-art bulk data management. The systems must provide the ability to securely transfer 14 mega bytes of data from one center to another within the network or concurrently transfer to several centers in a maximum of 30 minutes [JOPES ROC 1983, 104]
- o. Data system monitor for reconfiguration and priorities [JOPES ROC 1983, 105; JCS Pub 6-03.10 Vol II, Annex B, B-6]
- p. Interactive processing [JOPES ROC 1983, 105]
- q. Stringent protocols will provide disciplined high-speed atomic information exchange between nodes [JOPES ROC 1983, 13]
- r. Partial data degradation via decentralized databases and multiple exchange routes [NIS ROC 1983, 6, 20]
- s. Direct and immediate access to databases by Action Officers [NIS ROC 1983, 10]

- t. DBMS to accommodate distributed databases with access to various databases transparent to users [NIS ROC 1983, 25, 48]
- u. Assured interoperability between NMCS and major operational command centers [NIS ROC 1983, 4]
- v. Data capture at place of transaction and close to source as possible [JOPES ROC 1983, 110]
- w. Automated group situation display capability [JOPES ROC, 106; NIS ROC 1983, 7]

H.4.2.3 Network Services

The WAM network service includes both communications, requisite connectivity interfaces, automatic message handling, and, most important, interoperability. The following capabilities are needed to satisfy all network service requirements.

- a. System accounting and performance monitoring [JOPES ROC 1983, 105]
- b. Operations over satellite relay, commercial telephone, fiber optics, analog and digital circuitry, common user networks [JOPES ROC 1983, 101-102]
- c. Connectivity to theater and supporting levels [JOPES ROC 1983, 21]
- d. Survivable communications [JOPES ROC 1983, 40; NIS ROC, 47]
- e. Airborne ADP for all JCS and unified and specified command. Airborne Command Posts (ABNCPs) require ADP support to allow timely warning, force employment, and management decisions. The ABNCP ADP systems will interface through available communications systems to securely obtain, process, and display timely data [JOPES ROC 1983, 108; JCS Pub 6-03.10 Vol II Annex B, B-8]
- f. Deployable ADP JOPES for a capability that will include a number of small deployable, lightweight, easy-to-use terminals and minicomputers to support Joint Tactical Forces (JTFs), subordinate commands, and units operating in a forward area to support status reporting, planning, and force deployment and employment [JOPES ROC 1983, 108]
- g. Portable terminals for forward areas [JOPES ROC 1983, 108,109; JCS Pub 6-03.10 Vol II, Annex B, B-9]

- h. Transportable automated display systems [USCENTCOM C3 Master Plan Vol I 1989]
- i. Communication concentrators at selective locations with limited and stand-alone processing and storage capabilities in order to reduce system cost and to enhance system reliability and survivability [JOPES ROC 1983, 109; JCS Pub 6-03.10 Vol II Annex B, B-8]
- j. Video display of C2-related data [JOPES ROC 1983, 109]
- k. Congruent communications interfaces [NIS ROC 1983, 25]
- l. Voice input and output capabilities to support selected functional areas in time-sensitive situations. The terminals should be small, lightweight, secure, deployable, and provided with crypto-coupled dial-up capability to be utilized with commercial or military telephones to support small, rapidly deployable units in forward areas during crisis situations and deployments. The terminal shall be equipped with a preformatted text message input capability with error checking [JOPES ROC 1983, 40; JCS Pub 6-03.10 Vol II Annex B, B-7]
- m. Capability to display and transfer a working color map between two or more headquarters [NIS ROC 1983, 8]
- n. On-line data communications capable of receiving, processing, storing, recalling, formatting, displaying, transmitting, and assisting in the generation and staffing of properly classified messages in a number of different formats, particularly in the standard or accepted DoD and NATO formats. This capability will have to support both the fixed and mobile command centers and should interface with major communications systems. Diverse routing will be routinely provided. Commonality of communications security equipment to provide terminal-to-terminal, terminal-to-host, and host-to-host protection of information is needed in order to provide for interactive operation [NIS ROC 1983, 43; JCS Pub 6-03.10 Vol II Annex B, B-5].
- o. WWMCCS interoperability with U. S. Tactical and NATO automated systems [USCENTCOM Master Plan Vol I June 1989; NIS ROC 1983, 44]
- p. When USEUCOM Tactical forces are operating outside the NATO area, they may be assigned to a JTF whose C3 systems must interface with the rest of USCINCEUR C3 systems and with WWMCCS [USEUCOM C3 Master Plan Vol II 1987].

- q. Interoperability with the ability for two or more systems to securely exchange information or services directly and satisfactorily between themselves and with their users is required. Interoperability includes the ability to interface with systems external to WWMCCS such as major tactical, NATO, State Department, and NCA systems. [JCS Pub 6-03.10 Vol II, B-7]
- r. Connectivity/Interfaces with
 - 1) Reserve Components (RC) to track unit activation, stationing, and schedules for movement from home station to mobilization station and embarkation [JOPES ROC 1983, 23]
 - 2) DoD service sustainment systems [JOPES ROC 1983, 23]
 - 3) National politico-military systems used for monitoring and assessing threats, passing NCA guidance, status of U. S. citizens and property in crisis areas [JOPES ROC 1983, 20-22]
 - 4) Defense Communication System (DCS) [JOPES ROC 1983, 101]
 - 5) NATO Integrated Communications System [JOPES ROC 1983, 101]
 - 6) Support systems and executive aids such as NMCS Display and Information Distribution System, the NMCS Information and Display System, the SAC Automated Command and Control System, the PACOM Crisis Action Information Distribution System, the Data Readout and Display System, and the User Automatic Message Handling System. [JOPES ROC 1983, 109; JCS Pub 6-03.10 Vol II Annex B, B-7]
 - 7) Congruent database structures and reporting systems [NIS ROC 1983, 1, 25]
 - 8) All Joint Staff Directorates [NIS ROC 1983, 5]
 - 9) Unique C2 Systems of DoD and other federal agencies, NATO, allies and other functional ADP support systems [JCS Pub 6-03.10 Vol II Annex B, A-8]
 - 10) CINC information systems [NIS ROC 1983, 44]
 - 11) Major WWMCCS commands [NIS ROC 1983, 44]
 - 12) Service and Agency Headquarters [NIS ROC 1983, 44]
 - 13) NATO [NIS ROC 1983, 44; JOPES ROC 1983, 97]

- 14) Major tactical systems and JTFs [JCS Pub 6-03.10 Vol II, I-3, Vol III, III-2]
 - 15) Connectivity between CINC and components] JCS Pub 6-03.10 Vol II, I-5]
 - 16) Combined Forces Command (Korea) [JCS Pub 6-03.10 Vol II, I-3]]
 - 17) Department of State [JOPES ROC 1983, 97]
 - 18) Environment Services [JCS Pub 6-03.10, Vol I, I-7]
 - 19) C2 ADP systems [JCS Pub 6-03.10 Vol II, Annex B, A-8]
- s. Network system monitoring capability will provide the ability to monitor system activities and provide the capability to rapidly cut off users during Priority or degraded operations and for security reasons. The sites must be capable of dynamically and rapidly reconfiguring the system during periods of outages or Priority operations [JCS Pub 6-03.10 Vol II Annex B, B-6]
 - t. Interconnect the HQ FORSCOM Information System (FIS), the FORSCOM C2 System (FC2S) and the FORSCOM WWMCCS host computer [FORSCOM C2 Master Plan, 3-2]

H.4.2.4 Operating System Services

- a. Selective printing of output
- b. System accounting and performance monitoring [JOPES ROC 1983, 105]
- c. High-density quick storage medium such as video disc capable of displaying reference documents and maps in a real-time mode [NIS ROC 1983, 9]
- d. Automatic system regeneration [JOPES ROC 1983, 40]
- e. System monitoring to provide the ability to monitor system activities and provide the capability to rapidly cut off user during Priority or degraded operations and for security reasons. The sites must be capable of dynamically and rapidly reconfiguring the system during periods of outages or Priority operations [JOPES ROC 1983, 105; JCS Pub 6-03.10 Vol II Annex B, B-6]
- f. Task scheduling
- g. Remote procedure call
- h. File management
- i. Review management

- j. Name server directory service
- k. System must have centralized clocking and synchronization capability [JOPES ROC 1983, 105]
- l. Action prompting [JOPES ROC 1983, 107, 108; JCS Pub 6-03.10 Vol II Annex B, B-8]

H.4.2.5 Programming Services

- a. Provide [at all responsibility levels] timely crisis action decision tools that are also separate from deliberate planning [JOPES MNS 1989; JOPES ROC 1983, 3, 12]
- b. High order languages (HOL) (Ada is mandated) [NIS ROC 1983, 48]
- c. Software for environmental support system (ESS) must be modular in design with top down logic and COBOL 74 compatible language [NIS ROC 1983, 13]
- d. Action Officer assignments by video
- e. Analysis programs integrated with functional types of processing such as database management, message processing, display control processing, communications interface control and processing [NIS ROC 1983, 7; JOPES ROC 1983, 57]
- f. System design must allow for growth in data processing capacity [NIS ROC 1983, 6]
- g. Software transportability [JOPES ROC 1983, 97, 98; JCS Pub 6-03.10 Vol II, Annex B, B-4]
- h. Expert system capabilities
- i. Use of commercial off-the shelf (COTS) software and nondevelopmental items (NDI) [WAM DCP; JOPES ROC 1983, 68, 69]
- j. Reuse software libraries
- k. Programming development environment/Tailored ADP tools [JOPES ROC 1983, 66]
- l. Software distribution
- m. Configuration management

- n. Data analysis [JOPES ROC 1983, 65]
- o. Real-time algorithms for "what if" simulation analysis [JOPES ROC 1983, 103; JCS Pub 6-03.10 Vol I Annex K, III-8]
- p. Track action requests from receipt to closure regarding whose action, change in status, comments on coordination [NIS ROC 1983, 9, 10]
- q. Word processing capability [NIS ROC 1983, 50; JOPES ROC 1983, 108]
- r. Change map features, post symbols, add a zoom capability [NIS ROC 1983, 8]

H.4.2.6 Security

- a. Full multilevel security to include intelligence information [JOPES ROC 1983, 40, 113]
- b. Flexibility to develop incremental security solutions
- c. Secure communications to DCS and NATO for both digital and analog information
- d. Close-hold planning capabilities [JOPES ROC 1983, 106; NIS ROC 1983, 51]
- e. WWMCCS ADP must be secure from unauthorized access, data manipulation, denial of service, and data retrieval [JCS Pub 6-03.10 Vol II Annex B, B-1; NIS ROC 1983, 50]

H.4.2.7 User Interface

- a. Man-machine interface to have easy-to-use input and output devices and English-like query languages, for use by non-ADP-trained personnel, that require no more than eight hours of training. Interactive tutorials will be available at two levels—detailed for the novice and compressed for the experienced user. Sign-on and subsequent queries or responses must be human engineered for ease of operation commensurate with security requirements. User work stations will be designed to provide the non-ADP-training functional user with all the necessary capabilities at one location [JOPES ROC 1983, 98; JCS Pub 6-03.10 Vol II Annex B, B-4]
- b. Color graphics capability [NIS ROC 1983, 8, 36, 53]

- c. ADP to provide graphic aids [USCENTCOM 1989]
- d. Function selection from menu-styled lists by separate touch-sensitive video panel or "mouse" [NIS ROC 1983, 36]
- e. Pictorial function representations instead of keyboard text [NIS ROC 1983, 36]
- f. Interactive processing and query/response [JOPES ROC 1983, 101, 105; JCS Pub 6-03.10 Vol II Annex B, B-50]
- g. Data presentation [JOPES ROC 1983, 65]
- h. Interactive decision aids through automated ADP [USCENTCOM 1989]
- i. Decision aids to determine effective weapons mix [USCENTCOM 1989]
- j. Present graphic displays on large screens, multiple small desk management stations, hard copy printers, and photographic-quality transparence copiers [NIS ROC 1983, 10]
- k. Change map features, post symbols add a zoom capability [NIS ROC 1983, 8]
- l. Display interfaces for information presentation on specific information support functions and subfunctions [NIS ROC 1983, 8]
- m. Capable of multicolor texts, maps, graphics displays [NIS ROC 1983, 8]
- n. Screen partitioning/splitting, multiple screens or both [NIS ROC 1983, 8]
- o. Three-dimensional displays [JOPES ROC 1983, 40]
- p. Graphics terminals [JOPES ROC 1983, 106]
- q. Computer graphics conferencing [JOPES ROC 1983, 106]
- r. Large multicolor group briefing displays [JOPES ROC 1983, 106]
- s. Briefing support to produce classified color viewgraphs and hard copy outputs to assist in briefing decision makers and their staffs. The system should provide integrated text editing, word processing, and graphic aids plus automatic data access and formatting. Photograph-quality graphic aids are the goals. Further, most command centers will require (for briefing support and crisis/force monitoring) multi-color group display and/or large wall screen capabilities for presenting large amounts of complex classified data for simultaneous review by a number of people. [JOPES ROC 1983, 106]
- t. Voice input/output [JOPES ROC 1983, 40, 106]

- u. Creation and management of U. S. Joint Task Force [USCENTCOM 1989; JCS Pub 6-03.10, Vol II, I-3]
- v. Action prompting with automated checklists, procedures, instructions, and decisions aids required at selected work stations to assist command post and battle staff personnel. Two levels of prompting will be provided at user option—one for the novice and one for the experienced user. [JCS Pub 6-03.10 Vol II Annex B, B-8; JOPES ROC 1983, 107, 108.]

Table H-5. Summary of Information Requirements for the Supported Theater Level (Level II), Part 1

Information Area	Information Categories	Level of Detail Appropriate to This Level
Forces		
U. S. Units and Unit types	U. S. conventional Force/Unit status	Database of unit type
U. S. force modules		Database of force modules
Readiness status		Summary readiness at major forces by region, significant problems
Allied Forces: status and location	Intelligence	Summary data
Hostile forces: action and location	Reconnaissance	Summary data
Supplies		
By classes of supplies, e.g., POL, ammunition, food	Logistics	Summary data—worldwide and regional supply and demand
Critical and pacing items	Critical material	Status of items subject to intensive monitoring
Equipment		
Airlift and other aircraft	Major equipment	Operational characteristics
Sealift	Type movement characteristics	Summary data—world wide and regional supply and demand (current and projected)
Tanks, tracked vehicles, wheeled vehicles		
Weapons systems	Critical material	Status of selected items subject to intensive monitoring
C ³ equipment (including EW and C ³ CM)	C ³	Status of C ³ systems
Plans		
Plan summaries, OPLAN, CONPLAN force lists	OPLAN/CONPLAN	Plan summaries, indexed by keywords, regions, type, and plan, etc.

Continued on next page

Table H-5. Continued.

Information Area	Information Categories	Level of Detail Appropriate to This Level
Plans (continued)		
Force closure estimates	Closure estimates	Summary data on force lists for approved plans (not necessarily detailed plans)
Flow plans, movement tables	Flow plans	Aggregate flow plans and movement schedules
Non-combatant evacuation (NEO) plans	NEO	Summary of NEO by region
Mobilization (MOB) plans	MOB	Summary of MOB requirements
Treaties and Agreements Including Host Nation Support		
Significant Events	Significant events, intelligence, reconnaissance	Detailed information for assessment
Execution Reports		
Status of employment, mobilization or deployment		Summaries, unless detail is requested
Environmental Data		
Weather forecast	Environmental data, intelligence, reconnaissance	Summary worldwide and regional forecasts
Climatology		Summary data on climate and terrain
Terrain		
C ³ Status	C ³	Summary worldwide detail on problems as necessary
NCA Guidance		
Reference Data		
Standard terms	Data dictionary/directory	Database of time, data elements codes, translations and data dictionary/directory for all data available within JOPES
Location orders GroupID2		
Data dictionary/directory		
History file		Historical reference file

Source: (JOPES ROC 1983, Annex B)

Table H-6. Summary of Information Requirements for the Supported Theater Level (Level II), part 1

Information Area	Information Categories	Level of Detail Appropriate to This Level
Forces		
U. S. Units and Unit types	U. S. conventional Force/Unit status	Database of units and organization in theater: location status, current activity, and readiness
U. S. Force modules		
Readiness		
Allied Forces: status and location	Intelligence	Relevant allied forces: same duties as U. S. organizations
Hostile forces: status and location	Reconnaissance	Hostile/other forces: order of battle, identify strength and location, activity in theater
Supplies		
By class of supply, e.g., POL, ammunition, food	Logistics Nuclear weapons status Chemical weapons status	Theater stockage levels, shortfalls, anticipated demand
		Chemical weapons status and requisitioned supplies (U. S. and allied support HNS, POMCUS, PWRS, WRSA, and other special categories)
Critical and pacing items	Critical material	Status of selected items of critical interest in theater
Equipment		
Airlift and other aircraft	Major equipment	
Sealift		
Tanks, tracked vehicles		
Wheeled vehicles		
Weapon systems		
	Type movement characteristics	Assets in theater (summary), operational characteristics, movement characteristics, anticipated demand and forecast supply
	Critical material	Status of critical items
C ³ equipment (including EW and C ³ CM)	C ³	Status of C ³ systems
Plans		
Plan summaries, OPLAN, CONPLAN force lists	OPLAN/CONPLAN	Regional plans prepared in aggregate or detailed planning force lists
Force closure estimates	Closure estimates	Force closure estimates

Continued on next page

Table H-6. Continued.

Information Area	Information Categories	Level of Detail Appropriate to This Level
Plans (continued)		
Flow plans, movement tables	Flow plans	Flow plan, movement tables
NEO plans	NEO	NEO location and status by country in region
Mobilization plans		Plans and schedules affecting selected regional plans
Treaties and agreements Including Host Nation Support	HNS	Applicable treaties and agreements, details where approp.
Significant Events	Events, intelligence, reconnaissance	Detailed information on events in region likely to affect regional responsibilities
Execution Reports		
Status of employment, mobilization or deployment		Detailed as required, otherwise summaries within region
Environmental Data		
Weather forecasts	Environmental data, intelligence, reconnaissance	Regional forecast in detail, summary worldwide forecast, detail on problems affecting regional operations
Climatology		
Terrain		
C³ status	C ³	Summary worldwide, detailed regional status
NCA guidance		As applicable in orders and directives from JCS
Reference Data		
Standard terms, location, codes and equivalents, data, data dictionary/directory		Database of terms, data elements, codes, translation and full data dictionary/directory for all data available in JOPES

Source: (JOPES ROC 1983, Annex B)

Table H-7. Summary of Information Requirements for the Supporting Level (Level III), part 1

Information Area	Information Categories	Level of Detail Appropriate to This Level
Forces		
U. S. Units and Unit types	U. S. conventional	Own units status, location, activity, summary data
U. S. Force modules	Force/Unit status	
readiness status		
Allied Forces: Status and location	Intelligence	N/A unless for specific support requirements
Hostile Forces: Status and location	Reconnaissance	N/A
Supplies		
By class of supply, e.g., POL, ammunition, food critical and pacing items	Logistic HNS Nuclear weapons status, critical material	Supply and demand data (as applicable to functional mission)
Equipment		
Airlift and other aircraft	Major equipment	
Sealift		
Tanks, tracked vehicles	Type movement characteristics	
Wheeled vehicles		Own resources (as applicable to functional mission)
Weapons systems		
C ³ equipment (including EW and C ³ CM)		
Plans		
Plan summaries, OPLAN, CONPLAN force list force closure estimates, flow plans, movement table	OPLAN/CONPLAN, census estimates, flow plans	Summary requirements (if applicable to functional mission), detailed if functional mission involves support to a plan, detailed if functional mission involves mobilization
NEO	NEO	
Mobilization plans	MOB	
Treaties and Agreements Including Host Nation Support	HNS	Nothing (unless HNS or allied logistics requirements)
Significant Events	Significant events, intelligence, and reconnaissance	As required to support action
Execution Reports		
Status of employment, mobilization or deployment		Detailed data (if these are applicable to functional mission)

Continued on next page

Table H-7. Continued.

Information Area	Information Categories	Level of Detail Appropriate to This Level
Environmental Data		
Weather forecasts	Environmental data,	General worldwide summary data,
Climatology	intelligence,	worldwide or regional detailed
Terrain	reconnaissance	data (if applicable to functional mission)
C³ status	C³	Summary status (if applicable to functional mission)
NCA guidance		As applicable in orders and directives from JCS
Reference Data		
Standard terms, location, codes and equivalent data dictionary/directory		Database of terms, data elements, codes, translation and full data dictionary/directory for data available in JOPES and data in internal systems that are

Source: (JOPES ROC 1983, Annex B)

REFERENCES*

- [AFSC 1988] Armed Forces Staff College. 1 July 1988. *AFSC Pub 1, The Joint Staff Officer's Guide*. National Defense University.
- [Akin 1990] Akin, W. E., C. J. Brickell, B. R. Brykcynski, J. N. Donis, S. H. Edwards, D. Heystek, R. J. Knapper, J. P. Pennell, D. A. Wheeler. 1990. *Assessment of the Development Program for the AN/BSY-2 Submarine Combat System*. Alexandria, VA: Institute for Defense Analyses. IDA Paper P-2355.
- [Alexander 1990] Alexander, Michael. 1990. It's a Whole New HDTV Ball Game. *Computerworld* 24/49 (December 3): 18.
- [Andleigh 1990] Andleigh and Prabhat. 1990. *UNIX System Architecture*. Englewood Cliffs, New Jersey: Prentice Hall.
- [ARTWG 1987a] Ada Runtime Working Group (ARTWG) for the Special Interest Group (SIG) Ada. October 1987. *A Framework for Describing Ada Runtime Environments*.
- [ARTWG 1987b] Ada Runtime Working Group (ARTWG) for the Special Interest Group (SIG) Ada. December 1987. *A Catalog of Interface Features and Options for the Ada Runtime Environment, Release 2.0*.
- [ARTWG 1988] Ada Runtime Working Group (ARTWG) for the Special Interest Group (SIG) Ada. October 1988. *Catalogue of Ada Runtime Implementation Dependencies*. LabTek Corporation.
- [ASD/C³I 1990] ASD/C³I. 9 July 1990. Memo to Director, DCA and Director, NSA.
- [Atkinson 1989] Atkinson, M. et. al. September 1989. *The Object-oriented Database System Manifesto*. Technical Report No. 30-89, GIP ALTAIR, LeChesnay, France.
- [ATTCIS 1989] NATO. Army Tactical Command and Control Information System Permanent Working Group (ATTCIS) Working Paper 25, Edition 1.9 (Review Draft). 9 July 1989. Brussels, Belgium: Su-

*Information about cited standards are found in the accompanying IDA Paper, P-2457 [Nash 1991].

- preme Headquarters Allied Powers Europe (SHAPE). ATTCIS Working Paper 25.
- [Bach 1986] Bach, Maurice J. 1986. *The Design of the UNIX Operating System*. Englewood Cliffs, New Jersey: Prentice Hall.
- [Baker 1989] Baker, T. P. and Alan Shaw. The Cyclic Executive Mode and Ada. *Real-Time Systems Journal* 1/1 (June).
- [BBN 1981] Bolt, Beranek, and Newman, Inc. (BBN). 1981. *Specifications for the Interconnection of a Host and an IMP*. BBN Report 1822.
- [Boehm 1988] Boehm, Barry. 1988. A Spiral Model of Software Development And Enhancement. *IEEE Computer* (May).
- [Boland 1989a] Boland, Frederick E., editor. 1989a. *Stable Implementation Agreements for Open Systems Interconnection Protocols*. Version 2, Edition 1. Based on the *Proceedings of the December 1988 National Institute of Standards and Technology OSI Implementor's Workshop*; authorized reprint of NIST Special Publication 500-162
- [Boland 1989b] Boland, Frederick E. 1989b. *Ongoing Implementation Agreements for Open Systems Interconnection Protocols: Continuing Agreements*. Volume 1/1. New York: IEEE. Based on the *Proceedings of the December 1988 National Institute of Standards and Technology OSI implementor's Workshop*; authorized reprint edition of NISTIR 88-3824-3.
- [Boland 1990] Boland, Frederick E., editor. 1990. *Stable Implementation Agreements for Open Systems Interconnection Protocols*. Version 3 Edition 1. A NIST authorized 1990 reprint edition of NIST Special Publication 500-177.
- [Boral 1990] Boral, Haran et al. 1990. Prototyping Bubba, a Highly Parallel Database System. *IEEE Transactions on Knowledge and Data Engineering* (March).
- [Borden 1989] Borden, B. S. 1989. Future High Performance 3-D Graphics Workstations Will Be Vector Processors. In *Proceedings of the NCGA '89, 10th Annual Conference and Exposition Dedicated to Computer Graphics, April 17-20, Philadelphia, PA, 1989*, 279-281.

- [Brown 1990] Brown, Ed, William A. S. Buxton, and Kevin Murtagh. 1990. Windows on Tablets as a Means of Achieving Virtual Input Devices. *Human-Computer Interaction—INTERACT '90*, D. Diaper et al., editors, 675-681.
- [Buzen 1973] Buzen, J. P. and U. O. Gagliardi. 1973. The Evolution of Virtula Machine Architecture. In *National Computer Conference, 1973*, 291-299.
- [Child 1990] Child, Jeffrey. 1990. Video Controllers Present Trade-off Between Standards and Performance. *Computer Design* (August 1): 97-100.
- [CICNet 1991] CICNet, Inc., CICNet DS-3 Working Group. June 1991. *High Performance Applications on CICNet: Impact on Design and Capacity*. Ann Arbor, MI: CICNet, Inc.
- [CINC C2] CINC C2 Master Plan.
- [Cohen 1990] Cohen, Brian. 26 October 1990. Institute for Defense Analyses. Personal communication.
- [Creveld 1985] Creveld, Martin van. 1985. *Command in War*. Cambridge, MA: Harvard University Press.
- [CSC 1985a] DoD Computer Security Center. 25 June 1985. *Computer Security Requirements. Guidance for Applying the Department of Defense Trusted Computer System Evaluation Criteria in Specific Environments*. Washington, D.C.: DoD Computer Security Center. CSC-STD-003-85.
- [CSC 1985b] DoD Computer Security Center. 25 June 1985. *Technical Rationale Behind CSC-STD-003-85: Computer Security Requirements. Guidance for Applying the Department of Defense Trusted Computer System Evaluation Criteria in Specific Environments*. Washington, D.C.: DoD Computer Security Center. CSC-STO-004-85.
- [DEC 1990] Digital Equipment Corporation (DEC). July 1990. POSIX Tracking Report. Volume 2, Issue 3.
- [Deitel 1983] Deitel, H. M. 1983. *An Introduction to Operating Systems*. Reading, Massachusetts: Addison-Wesley.

- [DeWitt 1990] DeWitt, David J. et al. 1990. The Gamma Database Machine Project. *IEEE Transactions on Knowledge and Data Engineering* (March).
- [DoD 1978] Department of Defense. June 1978. Department of Defense Requirements for High Order Computer Programming Languages, STEELMAN. Washington, D.C.: DoD.
- [DoD 1980] Department of Defense. February 1980. Department of Defense Requirements for Ada Programming Support Environments, STONEMAN. Washington, D.C.: U.S. DoD.
- [DoD Defense 1988] Office of the Secretary of Defense. 1988. DoD Defense Guidance. Washington, D.C.
- [DoDD 3405.1] Department of Defense. 2 April 1987. DoDD 3405.1, Computer Programming Language Policy. Washington, D.C.: DoD.
- [DoDD 5100.30] Department of Defense Directive Number 5100.30, December 2, 1971
- [Dyer 1990] Dyer, D. S. 1990. A Dataflow Toolkit for Visualization. *IEEE Computer Graphics and Applications* (July): 60-69.
- [Emerging 1990] Emerging Technologies Group, Inc. 1990. *Open System Solutions: An Analysis of Application Environments*.
- [Foley 1986] Foley, James et al. *Ada Foundation Technology. Volume VIII: Software Requirements for WIS Graphics Systems Prototypes*. Alexandria, VA: Institute for Defense Analyses. IDA Paper P-1983.
- [FORSCOM 1988] FORSCOM C² Master Plan, 1988.
- [Freedman 1985] Freedman, Roy. 1985. *Programming with APSE Software Tools*. New York, NY: Petrocelli Books.
- [Graham 1989] Graham, Marc H. May 1989. *Guidelines for the Use of the SAME*. Pittsburgh, PA: Software Engineering Institute. Technical Report CMU/SEI-89-TR-16.
- [Green 1989] Green, M. and C. Shaw. 1989. The Data Paper: Living in the Virtual World. In *Proceedings of the Conference on Graphics Interface '90, Halifax, Nova Scotia, 14-18 May 1990*, 123-130.
- [Grudin 1989] Grudin, J. 1989. The Case against User Interface Consistency. *Communications of the ACM* 32 (October): 1164-1173.

- [Hall 1990] Hall, J. Storrs. 21 February 1990. Personal communication.
- [Hearn 1986] Hearn, D. and M. Baker. 1986. *Computer Graphics*. Englewood Cliffs, N.J.: Prentice-Hall.
- [Hemrick 1982] Hemrick, Christine. 1982. The Internal Organization of the OSI Network Layer: Concepts, Applications, and Issues. *Journal of Telecommunication Networks* 1/3, (Fall). Reprinted in *Computer Communications: Architectures, Protocols, and Standards*, William Stallings, editor, 195-205.
- [Hurson 1989] Hurson, A. R. et al. 1989. *Tutorial on Parallel Architectures for Database Systems*. IEEE Computer Society.
- [IDA 1982] Institute for Defense Analyses. *Joint Information Systems—Improved Application for Crises Management (U)*. Alexandria, VA: Institute for Defense Analyses. IDA Report R-266, SECRET.
- [Intel 1989] Intel Corporation. 1989. Micro 2000. *IEEE Spectrum* (October).
- [ISAT 1990] ISAT. 3 August 1990. ISAT Summer study, Defense Information Infrastructure. Michael L. Dertouzos, Chairman. Woods Hole, MA.
- [Ishii 1990] Ishii, Hiroshi. 1990. TeamWorkStation: Towards a Seamless Shared Workspace. In *Proceedings of the Conference on Computer-supported Cooperative Work, October 7-10, 1990, Los Angeles, CA*, 13-26
- [JSC PUB 1-02] Joint Chiefs of Staff. *Department of Defense Dictionary of Military and Associated Terms*. 1989
- [JCS PUB 6-03.10] Joint Chiefs of Staff. *WWMCCS Objectives and Management Plan. Management of the WWMCCS*. 1978.
- [Jeremiah 1991] Jeremiah, David E. 11 April 1991. Statement before the Committee on Armed Services, United States Senate.
- [JOPES COO 1986] Joint Operations Planning and Execution System (JOPES) Concept of Operations, 1986.
- [JOPES MNS 1989] JCS Mission Need Statement (MNS) for JOPES, JCS, JROCM-056-89, 5 December 1989.
- [JOPES PD 1988] NCS Joint Operations Planning and Execution System (JOPES) Procedures Description. JOPES, 1988.

- [JOPES ROC 1983] JCS Joint Operations Planning and Execution System (JOPES) Required Operational Capability (ROC), 5 July 1983.
- [JROC Mem 1989] Joint Requirements Oversight Council Memorandum - 056-89. 5 December 1989.
- [Jurgen 1989] Jurgen, Ronald K. 1989. Chasing Japan. *IEEE Spectrum* 26 (October): 26-30.
- [Kroemer 1981] Kroemer, K. H. E. 1981. Engineering Anthropometry: Designing the Workplace to Fit the Human. In *Proceedings of the Annual Conference of the American Institute of Industrial Engineers, May 17-20, 1981*, 119-126.
- [Kroemer 1988] Kroemer, K. H. E. 1981. VDT Workstation Design. *Handbook of Human-Computer Interaction*, M. Helander, ed., 521-539. New York: North-Holland.
- [Krohn 1990] Krohn, Nico, and Ed Scannell. 1990. Microsoft Preps Multimedia Spec. *Info World* 12/48 (November 26): 1f.
- [Kruger 1983] Kruger, M. W. 1983. *Artificial Reality*. Reading, MA: Addison-Wesley.
- [Kuhn 1990] Kuhn, D. R. 1990. Briefing on X-Window System Standards Update. Presented at the Applications Portability Profile and Open Systems Environment Users Forum, U.S. National Institute of Standards and Technology (NIST), Gaithersburg, MD.
- [LANTCOM 1988] LANTCOM C³ Master Plan 1988.
- [Lenat 1989] Lenat, D. B. 1989. Ontological Version Knowledge Engineering. *IEEE Transactions on Knowledge and Data Engineering* (March).
- [Lidinsky 1990] Lidinsky, William P. 1990. Data Communication Needs. *IEEE Network* (March): 28-33.
- [Linton 1989] Linton, M.A., J. M. Vlissides, and P. R. Calder. 1989. *Composing User Interfaces with Interviews*. IEEE Computer (February): 8-22
- [Lydic 1990] Lydic, Matthew. The Design and Implementation of a Parallel Database System. PhD diss., University of Maryland, College Park, MD. Expected completion, December 1990.

- [McCormick 1987] McCormick, B. H., T. A. DeFanti, and M. D. Brown, eds. *ACM SIGGRAPH Computer Graphics* 21 (November).
- [Melendez 1987] Melendez, Wilfred A. and Erik Lorenz Peterson. 1987. The Upper Layers of the ISO OSI Reference Model (Part I). *Computer Standards and Interfaces* 5 (1986): 13-46. Reprinted in *Computer Communications: Architectures, Protocols, and Standards*, William Stallings, editor, 373.
- [Morris 1973] Morris, James H. Jr.. 1973. Protection in Programming Languages. *Communications of the ACM* 16 (January).
- [Nash 1991] Nash, Sarah H., Robert P. Walker, and Kevin J. Saeger. 1991. *A Survey of Technical Standards for Command and Control Information Systems*. Alexandria, VA: Institute for Defense Analyses. IDA Paper P-2457, Draft.
- [NCSC 1987] National Computer Security Center. 1987. *Trusted Network Interpretation*. Ft. George G. Meade, MD. NCSC-TG-005.
- [NIS ROC 1983] JCS Statement of Required Operational Capability, National Military Command System (NMCS) Information System (NIS) in Support of Conventional Operations (ROC). Washington, DC: Department of Defense, July 1983
- [NIST 1990] National Institute of Standards and Technology (NIST). *Open System Standards: A Federal Strategy*, Gaithersburg, Maryland: NIST, 1990.
- [NIST APP/OSE 1990] National Institute of Standards and Technology (NIST). November 1990. *Application Portability Profile APP: The U.S. Government's Open System Environment Profile*. Draft
- [NIST APP/OSE 1991] National Institute of Standards and Technology (NIST). January 1991. *Application Portability Profile APP: The U.S. Government's Open System Environment Profile*.
- [Norman 1983] Norman, D.A. 1983. Design Principles for Human-Computer Interfaces. In *Proceedings of CHI '83 Conf. Human Factors in Computing Systems*, Boston, MA. (Dec): 1-10.
- [OE 1990] Optical Computer: Is Concept Becoming Reality?, OE Reports, The International Newspaper of Optical and Optoelectronic Ap-

- plied Science and Engineering, SPIE, The International Society for Optical Engineering, March 1990.
- [Open Systems 1990] Open Systems Newsletter. 1990. 4/1-2 (January-February): 25-26.
- [OSTP 1990] The Federal High Performance Computing Program, Office of Science and Technology Policy, 8 September 1989.
- [Pennell 1990a] Pennell, James P. et al. September 1990. *An Assessment of Software Portability and Reusability for the WAM Program*. Alexandria, VA: Institute for Defense Analyses. IDA Paper P-2456, Draft.
- [Pennell 1990b] Pennell, James P. et al. 1990. *Use of Design-of-Experiment to Evaluate File Transfer Protocols*. Invited Paper, 1990 Joint Statistical Meeting, Anaheim, CA. August 1990.
- [Rand 1980] Rand Corporation. NATO Management: Peace-to Crisis Transition RAND R-2576, February 1980, SECRET.
- [Rosenbloom 1989] Rosenbloom, L. J. 1989. Scientific Visualization at Research Laboratories. *IEEE Computer* 22/8 (August): 68-101.
- [Ross 1990] Ross et al. FDDI—A LAN Among MANs. *Computer Communication Review*. 20 (Month): 16-31.
- [SC21 1991] ISO/IEC JTC1/SC21 Information Retrieval, Transfer and Management for OSI. *Work on Security within SC21*. New York: ANSI, Inc.
- [Schneider 1979] Schneider, M.L., R.L. Wexelblat, and M.S. Jende. 1979. Designing Control Languages from the User's Perspective. *Proceedings of the IFIP TC 2.7 Working Conference on Command Languages*, Berchtesgaden, West Germany. (10-14 September): 184-186.
- [Schneider 1982] Schneider, M.L. 1982. Ergonomic Considerations in the Design of Command Languages. In *Human Factors and Interactive Computer Systems, Proceedings of the NYU Symposium on User Interfaces*, Y. Vassiliou (ed.), New York. (May 26-28): 141-162.
- [SDNS 1989a] SDNS Protocol and Signalling Working Group, National Security Agency. 1989a. *Secure Data Network System (SDNS) Security Protocol 3 (SP3)*, Specification SDN.301, Revision 1.5.

- [SDNS 1989b] SDNS Protocol and Signalling Working Group, National Security Agency. 1989b. *Secure Data Network System (SDNS) Security Protocol 4 (SP4)*, Specification SDN.401, Revision 1.3.
- [SDNS 1989c] SDNS Protocol and Signalling Working Group, National Security Agency. 1989c. *Secure Data Network System (SDNS) Key Management Profile, Communication Protocol Requirements for Support of the SDNS Key Management Protocol*, Specification SDN.601, Revision.
- [SDNS 1989d] SDNS Protocol and Signalling Working Group, National Security Agency. 1989d. *Secure Data Network System (SDNS) Message Security Protocol (MSP)*, Specification SDN.701, Revision 1.5.
- [SDNS 1989e] SDNS Protocol and Signalling Working Group, National Security Agency. 1989e. *Secure Data Network System (SDNS) Directory Specifications for Utilization with the SDNS Message Security Protocol (MSP)*, Specification SDN.702, Revision 1.4.
- [SDNS 1989f] SDNS Protocol and Signalling Working Group, National Security Agency. 1989f. *Secure Data Network System (SDNS) Access Control Concept Document*, Specification SDN.801, Revision 1.3.
- [SDNS 1989g] SDNS Protocol and Signalling Working Group, National Security Agency. 1989g. *Secure Data Network System (SDNS) Access Control Specification*, Specification SDN.802, Revision 1.0.
- [SDNS 1989h] SDNS Protocol and Signalling Working Group, National Security Agency. 1989h. *Secure Data Network System (SDNS) Access Control Specification*, Addendum 1, *Access Control Information Specification (ACIS)*, Specification SDN.802/1, Revision 1.0
- [SDNS 1989i] SDNS Protocol and Signalling Working Group, National Security Agency. 1989i. *Secure Data Network System (SDNS) Key Management Protocol, Definition of Services Provided by the Key Management Application Service Element (KMASE)*, Specification SDN.902, Revision 3.2.

- [SDNS 1989j] SDNS Protocol and Signalling Working Group, National Security Agency. 1989j. *Secure Data Network System (SDNS) Key Management Protocol, Specification of the Protocol for Services Provided by the Key Management Application Service Element (KMASE)*, Specification SDN.903, Revision 3.2
- [SDNS 1989k] SDNS Protocol and Signalling Working Group, National Security Agency. 1989k. *Secure Data Network System (SDNS) Key Management Protocol, SDNS Traffic Key Attribute Negotiation*, Specification SDN.906, Revision 1.3b.
- [Shapiro 1990] Shapiro, Syd. 1990. Firms Collaborate on Mammoth Memory Systems. *Computer Design* (News Edition) (June 18).
- [SIGMOD 1989] Future Directions in DBMS Research: Laguna Beach Participants. *SIGMOD Record* 18 (March).
- [Silberschatz 1990] Silberschatz, A. M. Stonebraker, and J. D. Ullman. February 1990. *Database Systems: Achievements and Opportunities*. Austin, TX: University of Texas at Austin. TR-90-22.
- [Smith 1990] Smith, Dennis B. and Paul W. Oman. May 1990. Software Tools in Context. *IEEE Computer*.
- [Stankovic 1987] Stankovic, J. A. and K. Ramamritham. 1987. The Design of the Spring Kernel. In *Proceedings of the IEEE Real-Time System Symposium, Washington, D.C. December 1987*, 146-157.
- [Stankovic 1988] Stankovic, John A. Misconceptions about Real-time Computing. *IEEE Computer* (October): 10-19.
- [Stankovic 1989] Stankovic, J.A., W.A. Halang, and M. Tokoro. 1989. Editor's Editorial. *The Journal of Real-Time Systems* 1 (June).
- [Stonebraker 1989] Stonebraker, Michael. 1989. Future Trends in Database Systems. *IEEE Transactions on Knowledge and Data Engineering* (March): 33
- [Strassman 1991] Strassman, Paul. 24 April 1991. Testimony to the Defense Subcommittee of the House Appropriations Committee, U.S. House of Representatives.
- [Sung 1990] Sung, H. C. K., G. Rogers, and W. Kubitz. 1990. A Critical Evaluation of PEX. *IEEE Computer Graphics and Applications* 10 (November): 65-75.

- [Synder 1988] Synder, H. L. 1988. Image Quality. In *Handbook of Human-Computer Interaction*, 437-473. Elsevier Science Publishers.
- [Tanenbaum 1981] Tanenbaum, Andrew S. 1981. *Computer Networks*. Englewood Cliffs, New Jersey: Prentice Hall.
- [Tanenbaum 1987] Tanenbaum, Andrew S. 1987. *Operating Systems Design and Implementation*. Englewood Cliffs, New Jersey: Prentice Hall.
- [Tannas 1989] Tannas, Lawrence E. 1989. Status and Trends in Flat-panel Displays. *IEEE Spectrum*. 26 (October): 34-35.
- [Thomas 1989] Thomas, Ian. November 1989. PCTE Interfaces: Supporting Tools in Software Engineering Environments. *IEEE Software*.
- [Toole 1989] Toole, R. 1989. Xtici: A Device-independent Color Editor for the X Window System. In *Proceedings of NCGA '89, 10th Annual Conference and Exposition Dedicated to Computer Graphics, April 17-20, 1989, Philadelphia, PA*, 481-487.
- [TR 9007] TR 9007, *Information Processing Systems - Concepts and Terminology for the Conceptual Schema and the Information Base*. July 1987.
- [USCENTCOM 1989] USCENTCOM C³ Master Plan Volume I, 1989.
- [WAM DCP 1989] Defense Communications Agency (DCA). November 1989. *WWMCCS ADP Modernization (WAM) Decision Coordinating Paper (DCP)*. Washington, DC: DCA.
- [Ware 1988] Ware, C. and D. R. Jessome. 1988. Using the Bat: A Six-dimensional Mouse for Object Placement. *IEEE Computer Graphics and Applications* (November): 65-70.
- [Washington 1991] WashingtonTechnology. 13 June 1991. [Title needed.]
- [Watabe 1990] Watabe, Kazuo, Shiro Sakata, Kazutoshi Maeno, Hideyuki Fukuoka, Toyoko Ohmori. 1990. Distributed Multiparty Desktop Conferencing System: MERMAID. In *Proceedings of the Conference on Computer-supported Cooperative Work, October 7-10, 1990, Los Angeles, CA*, 17-38.
- [Webster 1985] Webster's Ninth New Collegiate Dictionary. 1985. Springfield, MA: Merriam-Webster Inc.
- [Whitton 1989] Whitton, M. C. 1989. Visualization Accelerators. In *Proceedings of NCGA 1989: 10th Annual Conference and Exposition*

Dedicated to Computer Graphic, April 17-20, Philadelphia, PA, 1989, 331-340. Also published in Handbook of Human-Computer Interaction, Helander, M., editor. 1988. NY: Elsevier Science Publishers.

[WWMCCS 1974]

WWMCCS and the JCS. 30 August 1974. FY 1963-1974: A Joint Staff Perspective of the Worldwide Military Command and Control System (U). Working Paper prepared by the WWMCCS Council Support Office, OJCS, SECRET.

ACRONYMS

2D	Two dimension(al)
3D	Three dimension(al)
4GL	Fourth Generation Language
ABNCP	Airbourne Command Post
ACC	Access Control Center
ACID	Atomicity, Consistency, Isolation, Durability
ACIS	Access Control Information Specification
ACL	Access Control List
ACP	Allied Communications Publications
ACPM	Application Control Protocol Machine
ACSE	Association Control Service Element
ADP	Automated Data Processing
ADS	Automated Data Systems
AD1	Addendum 1
AE	Application Entity
AEP	Application Environment Office
AGI	Active Group Integrity
AHWG	Ad Hoc Working Group
AI/ES	Artificial Intelligence and Expert Systems
AIS	Automated Information System
AMPS	Advanced Meteorological Processing System
ANA	Article Number Association
ANSI	American National Standards Institute, Inc.
APCI	Application Protocol Control Information
APDU	Application Protocol Data Units
API	Application Program Interface
APP	Application Portability Profile
APSE	Ada Programming Support Environment
ARPA	Advanced Research Projects Agency
ARTWG	Ada Runtime Working Group
ASCII	American Standard Code for Information Exchange

ASD	Assistant Secretary for Defense
ASE	Application Service Element
ASN	Abstract Syntax Notation
ASME	Association of Structural and Mechanical Engineers
ASN	Abstract Syntax Notation
ATCCIS	Allied Tactical Command and Control Information System
ATIS	A Tool Integration Standard
ATO	Air Tasking Order
AWIPS	Advanced Weather Interactive Processing System
BER	Basic Encoding Rules
BFE	BLACKER front end
BISDN	Broadband Integrated Services Digital Network
BLOB	Binary Large Object
BLOS	Beyond Line of Sight
C ²	Command and Control
C ² IE	Command and Control Information Exchange
C ³	Command, Control and Communications
C ³ CM	Command, Control, and Communications Countermeasures
C ³ I	Command, Control, Communications and Intelligence
CAD	Computer-aided Design
CAIS	Common Ada Programming Support Environment Interface Set
CALS	Computer-aided Acquisition and Logistics Support
CAM	Computer-assisted Manufacturing
CAP	Crisis Action Planning
CASE	Computer-assisted Software Engineering
CCIS	Command and Control Information System
CCTT	Consultative Committee for International Telegraph and Telephone
CCR	Commitment, Concurrency, and Recovery
CD-ROM	Compact Disk-Read only Memory
CDR	Commander
CGA	Computer Graphics Association

CGI	Computer Graphics Interface
CGM	Computer Graphics Metafile
CGRM	Computer Graphics Reference Manual
CIA	Central Intelligence Agency
CIE	Common Information Exchange
CIE-UCS	Common Information Exchange Uniform Chromaticity Scale
CIEL	Common Information Exchange Language
CIELUV	Common Information Exchange L*u*v*
CIM	Corporate Information Management
CINC	Commander-in-Chief
CL	Connectionless (Oriented)
CLNP	Connectionless Network Protocol
CLNS	Connectionless Network Services
CMIP	Common Management Information Protocol
CMIPM	Common Management Information Protocol Machine
CMIS	Common Management Information Service
CMISE	Common Management Information Service Element
CO	Connection Oriented
COA	Course of Action
COMPUSEC	Computer Security
CONPLAN	Concept Plan
CONS	Connection-oriented Network Services
CONUS	Continental United States
COTS	Commercial-off-the-shelf
CPE	Conventional Planning and Execution
CPEP	Commercial Product Evaluation Process
CPU	Central Processing Unit
CRF	Connection-related Functions
CRT	Cathode Ray Tube
CS-MUX	Circuit Switching Multiplexer
CSC-STD	Computer Security Center Standard
CSCW	Computer Support for Cooperative Work
CSD	Circuit Switched Data; Combat Suppoort Detachment; Character Sequence Detector

CSMA-CD	Carrier Sense Multiple Access with Collision Detection
DAA	Designated Approval Authority
DAC	Discretionary Access Control
DAD1	Draft Addendum 1
DAP	Document Application Profile
DARPA	Defense Advanced Research Projects Agency
DART	Deployable Automated Data Terminal Response Team
DAS	Dual Attachment Station
DAS/DM	Dual Attachment Station Dual Media Access Control
DAS/SM	Dual Attachment Station Single Media Access Control
DBM	Data Base Management System
DCA	Defense Communications Agency
DCE	Data Circuit-terminating Equipment
DCEC	Defense Communications Engineering Center
DCP	Decision Coordinating Paper
DCPS	Data Communications Protocol Standard
DCS	Defense Communications System
DCT	Discrete Cosine Transform; Digital Communication Terminal
DCW	Digital Chart of the World
DDI	Director, Defense Information
DDL	Data Definition Language
DDN	Defense Data Network
DEC	Digital Equipment Corporation
DES	Data Encryption Algorithm/Standard
DGIWG	Digital Geographical Information Working Group
DIA	Defense Intelligence Agency
DIB	Directory Information Base
DID	Data Item Description
DIF	Document Interchange Format
DIGEST	Digital Geographical Information Exchange Standard
DIS	Draft International Standard
DISA	Defense Information Systems Agency [formerly Defense Communication Agency (DCA)]

DISNET	Defense Integrated Secure Network
DIT	Directory Information Tree
DLA	Defense Logistics Agency
DMA	Defense Mapping Agency
DML	Data Manipulation Language
DMS	Defense Message System
DNA	Defense Nuclear Agency
DNS	Domain Name Server
DoD	Department of Defense
DoD-D	Department of Defense Directive
DoD-STD	Department of Defense Standard
DoDCSC	Department of Defense Computer Security Center
DoDIIS	Department of Defense Intelligence Information System
DODISS	Department of Defense Index of Specifications and Standards
DP	Draft Proposal
DPS-8	Distributed Processing System-8
DSA	Directory System Agent
DSP	Directory System Protocol
DSSA	Designated Systems Security Administrator; Distributed Secure Systems Architecture
DSSSL	Document Style Segmentation and Specification Language
DTD	Data Type Definition
DTE	Data Terminal Equipment
DTMP	Data Communications Protocol Standard Technical Management Panel
DTP	Distributed Transaction Processing
DUA	Directory User Agent
DVI	Digital Video Interactive
E ³	End-to-end Encryption
ECMA	European Computer Manufacturers Association
EDI	Electronic Data Interchange
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport

EDTV	Extended-Definition Television
EESP	End-to-end Security Protocol
EGA	Extended Graphics Adapter
EGP	Exterior Gateway Protocol
EIA	Electronics Industries Association
EISA	Extended Industry Standard Architecture
EMP	Electromagnetic Pulse
EPL	Evaluated Products List
ESPRIT	European Strategic Programme for Research in Information Technology
ESS	Environmental Support System
ETSI	European Telecommunications Standards Institute
EUCOM	European Command
EW	Electronic Warfare
FAA	Federal Aviation Administration
FADU	File Access Data Unit
FC ³ S	Forces Command, Control and Communications System
FCC	Federal Communications Commission
FCNTL	File Contol Function
FDDI	Fiber Distributed Data Interface; Fiber-optic Digital Data Inter-change
FEMA	Federal Emergency Management Agency
FFOL	Fiber Distributed Data Interface Follow-on Local Area Network
FIMS	Forms Interface Management System
FIPS	Federal Information Processing Standard
FIS	Forces Command Information System
FODA	Formal Office Document Architecture
FORMETS	(NATO) Message Formatting System
FOSI	Formatting Output Specification Instance
FRG	Force Requirements Generator
FTAM	File Transfer Access and Management
FTP	File Transfer Protocol
FTS	File Transfer Service

Gbps	Gigabits per second
GIS	Geographical Information System
GKS	Graphics Kernel System
GMAP	General Macro Assembly Program
GOSIP	Government Open Systems Interconnection Profile
GSA	Government Services Administration
GUI	Graphical User Interface
H-MUX	Hybrid Multiplexer
HDLC	High-level Data Link Control
HDTV	High-definition Television
HEL	Human Engineering Laboratory
HF	High Frequency
HNS	Host Nation Support
HOL	High Order Language
HP	Hewlett-Packard
HRC	Hybrid Ring Control
I&A	Identification and Authentication
I/O	Input and Output
ICA	Integrated Communications Architecture
ICASE	Integrated Computer-aided Software Engineering
ICCCM	Inter-client Communications Conventions Manual
ICMP	Internet Control Message Protocol
ID	Identification
IDA	Institute for Defense Analyses
IDN	Integrated Digital Network
IEC	International Electrotechnical Committee
IEEE	Institute for Electrical and Electronics Engineers, Inc.
IGES	Initial Graphics Exchange Specification
IP	Internet Protocol
IPC	Interprocess Communication
IRDS	Information Resources Dictionary System

IS	Intermediate System; International Standard
ISA	Industry Standard Architecture
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISP	International Standard Profile
IT	Information Technology
ITDN	Integrated Tactical-strategic Data Networking
ITSEC	Information Technology Security Evaluation Criteria
JBIG	Joint Bilevel Imaging Group
JCS	Joint Chiefs of Staff
JDC	Joint Deployment Command
JDS	Joint Deployment System
JIAWG	Joint Integrated Avionics Working Group
JTF	Joint Tactical Force
JOPES	Joint Operations Planning and Execution System
JOPS	Joint Operation Planning System
JPEG	Joint Photographic Experts Group
JROC	Joint Requirements Oversight Council
JSCP	Joint Strategic Capabilities Plan
JSPS	Joint Strategic Planning Systems
JSTARS	Joint Surveillance Target Attack Radar System
JTC	Joint Technical Committee
JTF	Joint Task Force
JTM	Job Transfer and Manipulation
KAPSE	Kernel Ada Programming Support Environment
Kbps	Kilobits per second
KDC	Key Distribution Center
km	Kilometer
KMAE	Key Management Application Entity
KMAP	Key Management Application Process
KMASE	Key Management Application Service Element
KMP	Key Management Protocol

LAN	Local Area Network
LAPB	Link Access Procedure-B of X.25
LF	Low Frequency
LLC	Logical Link Control
LOS	Line of Sight
LRM	Language Reference Manual
LSAP	Link Service Access Point
MAC	Mandatory Access Control; Media Access Control
MACF	Multiple Association Control Function
MAGTF	Marine Air Ground Task Force
MANPRINT	Manpower and Personnel Integration
MAPSE	Minimal Ada Programming Support Environment
Mbps	Megabits per second
MCEB	Military Communications Electronics Board
MERMAID	Multimedia Environment for Remote Multiple Attendee Interactive Decision-making
MHS	Message Handling System
MIB	Management Information Base
MIL-M	Military Manual
MIL-R	Military Requirement
MIL-STD	Military Standard
MIMD	Multiple Instruction Multiple Data
MLS	Multilevel Security
MNS	Mission Need Statement
MOB	Mobilization [plan]
MOTIS	Message-oriented Text Interchange System
MPC	Multimedia Personal Computer
MPDT	Multipeer Data Transmission
MPEG	Moving Picture Experts Group
MSP	Message Security Protocol
MTA	Message Transfer Agent
MTS	Message Transfer System

N/A	Not Applicable
NASA	National Aeronautics and Space Administration
NASP	Network Service Access Point
NATO	North Atlantic Treaty Organization
NBS	National Bureau of Standard
NCA	National Command Authorities
NCC	Network Control Center
NCCIS	NATO Command and Control Information System
NCS	Network Computing System; National Security Council
NCSC	National Computer Security Center
NGCR	Next Generation Computing Resource
NDI	Nondevelopmental Item
NEO	Noncombatant Evacuation Operation
NeWS	Network Extensible Window Management System
NFS	Network File System
NIDS	National Military Command Center Information for Decision-makers Systems
NIS	National Military Command System Information System
NIST	National Institute of Standards and Technology [formerly National Bureau of Standards]
NLSP	Network Layer Security Protocol
NMCC	National Military Command Center
NMCS	National Military Command Systems
NOAA	National Oceanic and Atmospheric Administration
NOPLAN	No Plan (operation/contingency)
NOSA	NATO Open Systems Interconnection Security Architecture
NP	New Project
NPDU	Network Protocol Data Unit
NRL	Naval Research Laboratory
NS/DS	Namespace & Directory Services
NSA	National Security Agency
NSAP	Network Service Access Point

NSDU	Network Service Data Unit
NSF	Network File System; National Science Foundation
NTCB	Network Trusted Computing Base
NTP	Network Time Protocol
NTSC	National Television System Committee
ODA	Office Document Architecture
ODF	Office Document Format
ODIF	Office Document Architecture and Interchange Format
ODL	Office Document Language
ODP	Open Distributed Processing
OIW	Open Systems Interconnection Implementors Workshop
ONC	Open Network Computing
OODA	Observation, Orientation, Decision, and Action
OPLAN	Operations Plan
OPORD	Operations Order
OPREP	Operation Report
OS	Operating System
OSD	Office of the Secretary of Defense
OSE	Open System Environment
OSF	Open Software Forum
OSI	Open Systems Interconnection
PABX	Automatic Branch Exchange
PAGODA	Profile Alignment Group for Office Document Architecture
PAR	Project Authorization Request
PBX	Private Branch Exchange
PC	Personal Computer
PCI	Protocol-Control-Information
PCIS	Portable Common Interface Set
PCTE	Portable Common Tool Environment
PCTE+	Portable Common Tool Environment, Secure Version
PDAD	Proposed Draft Addendum
PDE	Program Development Environment

PDES	Product Data Exchange Specification
PDL	Page Description Language
PDU	Protocol Data Unit
PHIGS	Programmer's Hierarchical Interactive Graphics System
PHY	Physical Layer Protocol
PICS	Protocol Implementation Conformation Specifications
PII	Protocol Independent Interface
PLP	Physical Layer Protocol
PMD	Physical Layer Medium Dependent
POL	Petroleum, Oils, and Lubricants
POMCUS	Prepositioning of Material Configured to Unit Sets
POSIX	Portable Operating System Interface for Computer Environments
PPBS	Planning, Programming, and Budgeting System
PS	Programming Services
PSE	Programming Support Environment
PSSG	Protocol Standards Steering Group
PSTN	Public Switched Telephone Networks
PSTP	Protocol Standards Technical Panel
PWG	Protocol Working Group
PWRS	Pre-positioned War Reserve Stocks
QoS	Quality of Service
RC	Reserve Component
RDA	Remote Database Access
REC	Radio Electronic Combat
RF	Radio Frequency
RGB	Red-Green-Blue
RI	Risk Index
RIB	Routing Information Base
RIP	Routing Information Protocol
ROC	Required Operational Capability
ROM	Read-only Memory

ROSE	Remote Operation Service Element
RPC	Remote Procedure Call
RSA	Rivest, Shamir, Adelman (algorithm)
RSRE	Royal Signals Research Establishment (UK)
RTS	Runtime System
RTSE	Reliable Transfer Service Element
RUM	Resource and Unit Monitoring
SAC	Strategic Air Command
SACF	Single Association Control Function
SAME	SQL Ada Module Extensions
SANISI	Security Architecture for NATO Information System Interconnection
SAO	Single Application Association Object
SAS	Single Attachment Station
SC	Standards Committee; Subcommittee
SCCS	Source Code Control System
SCIF	Secure Compartmented Information Facility
SDE	Secure Data Exchange
SDH	Synchronous Data Hierarchy
SDIF	Standard Generalized Markup Language Document Interchange Format
SDNS	Secure Data Network System
SDTS	Spatial Data Transfer Specification
SDU	Service-Data-Unit
SEE	Software Engineering Environment
SEI	Security Exchange Information
SEVMS	Secure VMS
SGML	Standard Generalized Markup Language
SHAPE	Supreme Headquarters Allied Personnel Europe
SIG	Special Interest Group
SILS	Standard for Interoperable Local Area Network Security
SITREP	Situation Report
SMF-PMD	Single Mode Fiber Physical Layer Medium Dependent

SMIB	Security Management Information Base
SMT	Station Management
SNDCF	Subnetwork Dependent Convergence Function
SNICP	Subnetwork Independent Convergence Protocol
SNISP	Subnetwork Independent Security Protocol
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Network
SORTS	Status of Resources and Training Systems
SP	Security Protocol; Service Provider
SPDL	Standard Page Description Language
SPDU	Session Protocol Data Unit
SPM	Synchronous Optical Network Physical Layer Medium Dependent
SQL	Structured Query Language [no longer an acronym but a word]
STANAG	NATO Standardization Agreement
STARS	Software Technology for Adaptable, Reliable Systems
STEP	Standard for the Exchange of Product Model Data
SWG	Special Working Group
T&E	Test and Evaluation
TC	Technical Committee
TCB	Trusted Computing Base
TCP/IP	Transmission Control Protocol/Internet Protocol
TCS	Trusted Communications Sublayer
TCSEC	Trusted Computer Security Evaluation Criteria
TDB	Trusted Data Base
TDI	Trusted Database Interpretation
TE	Terminal Equipment
TEK	Traffic Encryption Key
TELNET	Telecommunications Network
TFE	Transportation Feasibility Estimator
TFEL	Thin Film Electroluminescence
TIGER	Topologically Integrated Geographic Encoding and Referencing
TLCF	Teleconference

TLSP	Transport Layer Security Protocol
TNB	Trusted Network Base
TNI	Trusted Network Interpretation
TP	Transaction Processing; Transport Protocol
TPASE	Transaction Processing Application Service Element
TPDUs	Transport Protocol Data Unit
TPFDD	Time-phased Force and Deployment Data
TPSU	Transaction Processing Service User
TPSUI	Transaction Process Service User Invocation
TR	Technical Report
TRADACOMS	Working Party on Trading Data Communications
TRANSCOM	Transportation Command
TS	Top Secret
TSM	Tax System Modernization
TSS	Time Synchronization Service
TW/AA	Tactical Warning/Attack Assessment
UA	User Agent
UHF	Ultra-high Frequency
UI	User Interface
UIMS	User Interface Management System
UISRM	User Interface Services Reference Model
UK	United Kingdom
US	United States
USA	United States Army
USAETL	United States Army Corps of Engineers Engineering Topographic Laboratories
USAF	United States Air Force
USCENTCOM	United States Central Command
USCINCEUR	United States Commander in Chief, European Command
USCINCLANT	United States Command in Chief, Atlantic Command
USEUCOM	United States European Command
USFORSCOM	United States Forces Command
USGS	United States Geological Survey

USLANTCOM	U. S. Atlantic Command
USMC	United States Marine Corp
USN	United States Navy
USPACOM	U. S. Pacific Command
VGA	Video Graphics Adapter
VHF	Very High Frequency
VMS	Virtual Memory System
VPS	Vector Product Standar
VS	Virtual System
VT	Virtual Terminal
WAM	Worldwide Military Command and Control System Automated Data Processing Modernization
WAN	Wide Area Network
WBC	Wideband Channel
WD	Working Document or Working Draft
WES	Worldwide Military Command and Control System Entry System
WG	Working Group
WHSR	White House Situation Room
WIN	Worldwide Military Command and Control System Information Network
WINCS	Worldwide Military Command and Control System Intercomputer Network
WIS	Worldwide Military Command and Control Information System
WORM	Write Once, Read Many
WS	Work Specification
WWMCCS	Worldwide Military Command and Control System
XEU	Xerox Encryption Unit
XVT	eXtensible Virtual Toolkit

Distribution List for IDA Paper P-2490

NAME AND ADDRESS	NUMBER OF COPIES
Sponsor	
Mr. James Robinette JIEO/TVCF Defense Information Systems Agency Center for C3 Systems 3701 N. Fairfax Dr. Arlington, VA 22203	1 camera-ready copy
Other	
Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
IDA	
General Larry D. Welch, HQ	1
Mr. Philip L. Major, HQ	1
Dr. Robert E. Roberts, HQ	1
Ms. Ruth L. Greenstein, HQ	1
Mr. William A. Akin, CSED	1
Dr. Cy D. Ardoin, CSED	1
Mr. James Baldo, CSED	1
Mr. John M. Boone, CSED	1
Mr. Bill R. Brykcynski, CSED	1
Dr. David J. Carney, CSED	1
Dr. Brian S. Cohen, CSED	1
Ms. Anne Douville, CSED	1
Dr. Dennis W. Fife, CSED	1
Dr. Harlow Freitag, CSED	1
Dr. Karen D. Gordon, CSED	1
Ms. Deborah Heystek, CSED	1
Ms. Audrey A. Hook, CSED	1
Dr. Norman R. Howes, CSED	1
Dr. Richard J. Ivanetich, CSED	1

NAME AND ADDRESS	NUMBER OF COPIES
Mr. Robert J. Knapper, CSED	1
Mr. Steve Lawyer, CSED	1
Ms. Catherine W. McDonald, CSED	1
Mr. Terry Mayfield, CSED	1
Dr. Richard P. Morton, CSED	1
Ms. Sarah H. Nash, CSED	1
Ms. Katydean Price, CSED	1
Dr. Larry H. Reeker, CSED	1
Dr. Robert M. Rolfe, CSED	1
Prof. Edgar Sibley, CSED	1
Ms. Beth Springsteen, CSED	1
Mr. Stephen R. Welke, CSED	1
Dr. Richard L. Wexelblat, CSED	1
Dr. Craig A. Will, CSED	1
Dr. Robert I. Winner, CSED	1
Ms. Christine Youngblut, CSED	1
General Edward Bautz, SED	1
Prof. Thomas C. Bartee, SED	1
Mr. James Carlson, SED	1
General Gregory A. Corliss, SED	1
Dr. William L. Greer, SED	1
Dr. David L. Randall, SED	1
Dr. Robert P. Walker, SED	1
Mr. Philip J. Walsh, SED	1
Dr. Earl A. Alluisi, STD	1
IDA Control & Distribution Vault	3